Algebraic Varieties and System Design

Aabrandt, Andreas

Publication date:
2016

Document Version
Publisher's PDF, also known as Version of record

Technical University of Denmark

Andreas Aabrandt

Algebraic Varieties and System Design

PhD thesis, May 2016

DTU Electrical Engineering
Department of Electrical Engineering

Title: Algebraic Varieties and System Design
Titel: Algebraiske Varieteter og System Design

This dissertation was prepared by:
Andreas Aabrandt

Advisors:
Chresten Træholdt, Department of Electrical Engineering
Vagn Lundsgaard Hansen, Department of Applied Mathematics and Computer Science
Bjarne Poulsen, Department of Applied Mathematics and Computer Science

Center for Electric Power and Energy
Department of Electrical Engineering
Technical University of Denmark
Elektrovej, Building 325
2800 Kgs. Lyngby
Denmark

Abstract

Design and analysis of networks have many applications in the engineering sciences. This dissertation seeks to contribute to the methods used in the analysis of networks with a view towards assisting decision making processes. Networks are initially considered as objects in the category of graphs and later as objects in the category of hypergraphs. The connection with the category of simplicial pairs become apparent when the topology is analyzed using homological algebra. A topological ranking is developed that measures the ability of the network to stay path-connected. Combined with the analysis of cover ideals of hypergraphs, the topological ranking demonstrates the non-trivial decisions that needs to be considered in system design. All the methods developed here have an underlying common structure, namely that they all appear at solution sets for systems of polynomials. These solution sets are called algebraic varieties.

Resumé

Design og analyse af netværk har mange anvendelser i ingeniørvidenskaberne. Denne afhandling søger at bidrage til metoder til analysen af netværk og dermed til metoder der kan bruges til at støtte beslutninger. Der startes med at betragte netværk som objekter i kategorien af grafer, senere som objekter i kategorien af hypergrafer og deres samspil med kategorien af simpliciale par. Topologien undersøges ved hjælp af homologisk algebra, hvor der specielt udvikles en rangordning af elementer i netværk. Til sidst løses konkrete problemer ved at betragte overlejringsidealer for hypergrafer. Alle disse metoder har en underliggende struktur til fælles; de optræder alle som løsningsmængder for systemer af polynomiale ligninger, og netop disse løsningsmængder kaldes for algebraiske varieteter.

Preface

This dissertation is the result of work carried out at the Technical University of Denmark and supervised by Chresten Træholdt, Vagn Lundsgaard Hansen and Bjarne Poulsen. The work is funded by the Technical University of Denmark.

The main purpose of this dissertation was to investigate connections between agents acting in networks related to communication networks and power systems. The goal was to come up with new methods for analyzing networks while retaining a very flexible structure such that system ontologies were allowed to change over time. Another important objective was to build methods which would not exclude existing successful methodologies and frameworks.

Taking a view towards power systems, it became very clear that many different areas of mathematics and science are required and coexist in everything from design of power systems, to the communication systems currently in use, and not least to work within a political and sociological framework. These considerations were instrumental in the choice of taking a broad viewpoint as opposed to considering only one or two mathematical methodologies.

I owe special thanks and gratitude to Vagn, who has guided me through the world of mathematics for many years. I am particularly grateful for the opportunity to work closer with Vagn on projects of mutual interest in recent years; notably our discussions on topics in algebraic geometry and topology have proven to be very useful. I feel fortunate to have had the supervisor team consisting of Chresten, Vagn and Bjarne, who have all supported my efforts (and my un-traditional viewpoints) in developing theoretical tools for network analysis. Also, I am grateful to all my colleagues for making me feel right at home during my time at the university. Thanks to Chresten for the many inspiring conversations. Special thanks to Haoran for letting me use his LaTeX files for the IEEE bus systems.

Technical University of Denmark, May 2016

Andreas Aabrandt

Contents

1. Introduction **1**
- 1.1. Mathematical Modelling . . . 3
- 1.2. Control in Power Systems . . . 4
- 1.3. Equivalence Relations . . . 5
- 1.4. Vertex Covering of Hypergraphs . . . 5
- 1.5. Arithmetic in Finite Fields . . . 6
  - 1.5.1. Finding Subfields of Finite Fields . . . 8

2. Algebraic Varieties over Finite Fields **11**
- 2.1. Elementary Diophantine Geometry . . . 11
- 2.2. Polynomial Rings over Finite Fields . . . 14
- 2.3. Monomial Ordering . . . 15
- 2.4. Polynomial Division in $R[x_1, \ldots, x_n]$ . . . 16
- 2.5. Affine Varieties over Finite Fields . . . 18
  - 2.5.1. The Circle Equation over Finite Fields . . . 19
  - 2.5.2. Patterns in the Number of Solutions to the Circle Equation . . . 23

3. Elements of Categorical Algebra **27**
- 3.1. Categories . . . 27
- 3.2. Functors . . . 29
- 3.3. Natural Transformations . . . 30
  - 3.3.1. Some Results Related to Determinants . . . 32

4. Gröbner Bases and Their Applications **35**
- 4.1. S-polynomials . . . 35
- 4.2. Gröbner Bases . . . 36
  - 4.2.1. Computing Gröbner Bases . . . 37
- 4.3. Reduced Gröbner Bases . . . 37
  - 4.3.1. The Circle Equation Revisited . . . 38
- 4.4. Computing Intersection Ideals . . . 42
  - 4.4.1. Parallelized Intersection of Ideals . . . 43
- 4.5. Future Research: Applications to Packing Problems . . . 52
  - 4.5.1. Example of Bin Packing . . . 52

5. Applied Homology Theory **55**
5.1. The Category of Simplicial Pairs 55
5.2. The Category of Chain Complexes 56
5.3. Relative Homology 57
5.4. Modelling Networks with Simplicial Pairs 58
5.5. Topological Ranking 59
5.6. Applications of Topological Ranking 61
5.7. Future Research: Stanley-Reisner Rings and Koszul Complexes 64
5.7.1. Alternative Boundary Formula 65

6. Optimization of Network Design **67**
6.1. Edge and Cover Ideals of Hypergraphs 68
6.2. Phasor Measurement Unit Placement 70
6.2.1. Mandatory Placement 73
6.2.2. Mutual Exclusion Zones 74
6.2.3. IEEE 30 Bus System 76
6.3. Concluding Remarks 78

7. Discussion and Conclusion **79**

A. Haskell Code for Working with the Circle Equation **81**

B. Manuscript for Article: The Circle Equation over Finite Fields **83**

C. Computing Homology **93**

D. Code for Vertex Cover Calculations **95**

List of Figures

1.1. Vertex cover of a graph 6

2.1. The circle $\mathcal{V}(x^2 + y^2 - 1)$ 20
2.2. Algebraic variety defined by $x^2 + y^2 - 1 \in \mathbb{F}_{19}[x, y]$. 23

4.1. Comparison of algorithms for computation of cover ideals 45
4.2. Comparison of memory usage for computation of cover ideals 46
4.3. Random graph with intersection order not shuffled, i.e. ordered 47
4.4. The same random graph with intersection order shuffled 48
4.5. Comparison of parallel algorithms for random graphs 50
4.6. Comparison of all algorithms for random graphs 51

5.1. $\mathfrak{F}$ and the nine agents. 62
5.2. The nerve of $\mathfrak{F}$ and the rank of all elements. 62
5.3. Simplicial Complex 64

6.1. Hypergraph 68
6.2. IEEE 14 Bus System 70
6.3. IEEE 14 bus system topology 71
6.4. IEEE 14 bus system topology with outage at 4 and 6 72
6.5. IEEE 14 bus system topology with outage at 4 and 6 72
6.6. IEEE 14 bus system topology with outage at buses 4 and 6 73
6.7. IEEE 14 bus system topology with mandatory placement 74
6.8. IEEE 14 bus system topology with mutual exclusion zone 75
6.9. IEEE 30 Bus System 76
6.10. IEEE 30 Bus System with mandatory placement of PMUs 77

List of Tables

6.1. Possible minimal coverings of IEEE 30 bus system 77
6.2. Minimal coverings of IEEE 30 bus system with mandatory placement . . 78

Chapter 1

Introduction

The problems tackled in this dissertation involves the scientific disciplines of electrical engineering and algebraic geometry and topology. The main object of interest will be networks in a very broad sense, e.g., power grid topology or communication networks. Being both an electrical engineer and a very skilled mathematician, Solomon Lefschetz did combine his interests in electrical engineering and geometry and topology, towards the end of his life, and in a book [24] finished by his colleagues, Lefschetz describes the connection between Kirchhoff's laws and algebraic topology. As shall become apparent, there are many similarities between topology and electrical engineering. Widening the subject of topology to encompass geometry, will fully absorb theoretical methods used in electrical engineering.

One of the interesting bi-products of the development of algebraic topology, was the introduction of category theory. A theoretical framework for describing how theories can be related. In topology it originally related topological spaces and continuous maps between topological spaces to abelian groups and group homomorphisms; the relation was a set of numbers, which in terms of topological spaces is very difficult to calculate, while in terms of abelian groups and group homomorphisms, these numbers have become tangible and they are called Betti numbers, named after Italian mathematician Enrico Betti.

Category theory has become a subject with its own ontology and though we will not consider a fully modern category theory in this dissertation, we will consider just enough to make elementary use of it. The primary use in this dissertation will be to organize the mathematical modelling process and develop methods for supporting decision making in design of networked systems.

We focus on the theory of algebraic varieties over finite fields and homology theory with finite fields as coefficients. The first belong to the study of algebraic geometry and it is the study of roots of polynomials. Over finite fields, polynomials behave very differently than over fields of characteristic zero, e.g., $\mathbb{Q}$, $\mathbb{R}$ and $\mathbb{C}$. One of the advantages of doing calculations over finite fields, is the efficiency of representing such structures on a computer as opposed to working with real numbers, which are uncountable. The second subject that we shall cover is part of algebraic topology and it is essentially concerned with studying the shape of mathematical objects. These methodologies will

be developed and prepared such that they can be applied to solve problems in the vaguely defined notion of networks. By allowing for a class of ontologies rather than singling out a discipline, our methods can be used to solve a wide variety of problems, ranging from problems in pure mathematics to both well-known and undefined problems in power and energy engineering.

The study of polynomial rings and their ideals have a long history. The use of polynomials in geometry can be dated back to the time of old Mesopotamia, and polynomials can be seen lurking in the background of ancient Greek geometry in Euclid's Elements [14] from about 300 BC. Jumping many years ahead, we have the birth of modern algebra and with it, a structural way of working and organizing the theory of polynomials. Many problems can efficiently be handled by studying polynomial ideals in polynomial rings.

One of the problems when working with ideals in polynomial rings is that often it is not exactly clear whether or not a given polynomial in a polynomial ring belongs to a given ideal. This problem is called the ideal membership problem and it was not until Bruno Buchberger in 1965, in his Ph.D. thesis [8], developed the theory of Gröbner bases, also called standard bases, that it has become possible to solve this problem conveniently. The idea of Gröbner bases was anticipated prior to Buchberger's dissertation, but it was Buchberger who gave it a formulation apt for computation.

In recent years, Gröbner bases have grown to be a useful tool in applied mathematics, mainly stimulated by the development of modern personal computers. The abstract nature of Gröbner bases have once again demonstrated that pure and applied mathematics belong together. Many areas of science and engineering are now increasingly realizing the positive results obtained by using computer algebra systems to solve problems in their respective fields.

Among the very interesting applications of Gröbner bases we mention their successful generalization of already established methods, such as Gaussian elimination in linear algebra and the simplex method from operations research. Most strikingly is the potential it has in connection with research in artificial intelligence, were Gröbner bases and generalizations of them are being used in automatic theorem proving and reasoning systems. For an introduction to automatic geometric theorem proving, see [9].

Lately there has been an increasing interest in the connection between suitably chosen monomial ideals and their applications to graph theory. The concepts of *edge ideals* and *cover ideals* have emerged and have given a new and interesting perspective to already known problems in graph theory.

The introduction to homology theory given here is very classical and follows standard introductions to the subject. We are interested in the connection it has with the structure of networks and we develop a ranking of elements in a network at the end of Chapter 5. The ranking, which we call *topological ranking*, determines how "critical" a set of elements in a network is, to the overall ability of the network to stay path connected, i.e. in one piece. The information that this ranking yields is fundamental and cannot be changed once a shape, i.e. topology, is fixed. For dynamic networks, the topological ranking provides information about which collections of elements are most important if we want the connectedness to stay intact. This is often sought in power grids or

communication networks and therefore we find the topological ranking interesting.

At the end of Chapter 4, a new algorithm is supplied for computing cover ideals, which solves the (exact) graph covering problem adaquately and in a scalable way. In Chapter 6 the methods from Chapter 4 and Chapter 5 are combined to make analysis of some basic IEEE bus systems.

1.1. Mathematical Modelling

Most of engineering relies on mathematical modelling techniques. The modelling process is seldom rigorous and most often it is considered artistic rather than scientific. In order to put the mathematical modelling process on a rigorous basis, we need to first consider some of the typical methods used in modelling engineering systems. The most widely applied areas of mathematics used in engineering today are functional analysis and global analysis. The methods from the aforementioned areas of mathematics are extremely effective for a large group of engineering problems. However as with most things there are obvious limitations of the continuous methods when considering problems which exhibit discrete phenomena.

Another very popular collection of mathematical methods are those solving mathematical programming problems, e.g., linear programming and dynamic programming etc. These methods are very effective at solving discrete optimization problems once the correct formulation of a given problem is proposed. Also model predictive control is a widely used, yet theoretically immature, approach often taken in control engineering.

Whenever it is necessary to count objects in a system, one often consider combinatorial methods, with the modelling process very obscured. If one considers topological methods to count objects, then the arithmetic is a priori defined by the Euler characteristic or more generally by the Lefschetz trace formula. The advantage of using topology lies in the flexibility to count more complicated or even specific objects.

Category theory, also an invention from algebraic topology, is instrumental in tackling deep philosophical issues in modelling modern systems and it will serve at the foundation of all modelling provided in this thesis.

Defining relationships is a common approach to modelling. These relationships are frequently implicit and delicate assumptions often needs explanation. For instance, a graph can be constructed to illustrate connections between certain objects. This could be a social network where vertices are people and edges represent people being friends or linked to each other.

The kind of model chosen is intimately linked with the problem whose solution is sought. A common type of problem solving technique is to divide the original problem into sub-problems and solving these while inferring the sub-solutions to back to a solution of the original problem. Such methods are utilized in dynamic programming and many other problem solving paradigms.

In pure mathematics many problems are solved by constructionist approaches. These differ from the aforementioned approach by disguising the solution in an equivalence between two seemingly different problems. One is the original problem and the other,

a much easier problem by some appropriate measure. For instance the case of vertex covering of a graph. A vertex covering of a graph is a subset of vertices in a given graph such that for each edge in the graph, at least one of the vertices that the edge connects, is in this subset. Many algorithms exists that solves this rather difficult problem, and the solution is usually found via optimization means, i.e. dividing the problem into subproblems, e.g., see [20] or for a general introduction to dynamic programming see [5, 6]. The problem can, however, be stated in another way, which does not directly relate to graphs. The (minimum) vertex covering problem can be transformed into a problem of calculating intersections of monomial ideals, also called cover ideals, which we shall discuss in Chapter 6.

1.2. Control in Power Systems

Systems and control are particularly important in power systems, both in design and operation. The stability of electric power systems have become crucial in our modern society as there are today many systems depending on a steady supply of power for their day-to-day operations. Control systems are employed on all levels in all power stations, transformer stations etc. Most of these control systems are interconnected and rely on feedback. The feedback systems must work together in order to meet demand by regulating production. But also, there are perversions in the networks or local stations in the form of equipment failure or similar disturbances; situations that must be dealt with whether the systems are *human in the loop* or not.

Today, most if not all electricity is distributed using alternating current. The advantage is that voltage levels can be changed without major power losses. This is done via the transformer stations. The disadvantage is that generators must be synchronized to the voltage variations and this in turn means synchronizing rotors in generators. It is quite clear that synchronizing rotors across a large network is no small feat.

Among the control strategies are dividing a network into subnets, switching off parts of the network, lowering voltage levels etc. Here topology of the network, e.g., power grid, is very important and simulation-based methods can test if planned changes to the topology effect the operation of the power grid. For these purposes there are a number of available power grid topologies, which can be analyzed and used for testing, e.g., the IEEE bus systems or using randomly generated topologies, see [34].

1.3. Equivalence Relations

The theory of rings, modules and fields have become unavoidable in modern mathematics and in most applied fields ranging from systems theory and control theory to computational biology. Their applications range across all areas of mathematics and electrical engineering. Some of the fundamentals which will prove very useful in many applications will be briefly introduced in this section. We assume that the reader is familiar with groups and rings.

The notion of equivalence relations is a very fundamental and important idea in mathematics. It serves as a means to define what is meant by equivalence of abstract mathematical structures. Recently they have been shown to facilitate the mathematical modelling process in engineering.

In short, an equivalence relation is a relation which defines when two or more elements are the same according to some predefined criteria. An equivalence class then consists of all the elements which are the same in the aforementioned sense.

More precisely,

Definition 1. A relation $\sim$ is called an *equivalence relation* if it is reflexive, symmetric and transitive, i.e. for a given set S, a relation $\sim$ where, for all $a, b, c \in S$, it holds that

1. $a \sim a$ (reflexive)
2. $a \sim b$ if and only if $b \sim a$ (symmetric)
3. $a \sim b$ and $b \sim c$ implies $a \sim c$ (transitive).

An equivalence relation induces a collection of equivalence classes. When modelling one often first identifies the desired equivalence classes and then look for an equivalence relation which induces exactly these equivalence classes.

1.4. Vertex Covering of Hypergraphs

Graphs is one of the most useful tools in mathematics and we present only some very elementary definitions which we shall use in this thesis.

Definition 2. A *hypergraph* $H = (V, E)$ consists of a vertex set $V = \{v_1, \ldots, v_s\}$ and a hyperedge set $E = \{e_1, \ldots, e_t\}$, where each hyperedge e_i is a set of vertices from V.

Clearly a hypergraph is a generalization of a graph and we have

Definition 3. A *graph* $G = (V, E)$ is a hypergraph, where for each $e \in E$ there are exactly 2 vertices in e, an initial vertex and a terminal vertex.

We shall later discuss the connection between graphs and hypergraphs, when we introduce homology and also when we introduce square-free monomial ideals used to analyze the phasor measurement unit placement problem.

Definition 4. Given a hypergraph $H = (V, E)$, with vertex set V and hyperedge set E. A *vertex covering* is a subset $C \subset V$ where for each $e \in E$ there exists a vertex $v \in C$ such that v belong to e.

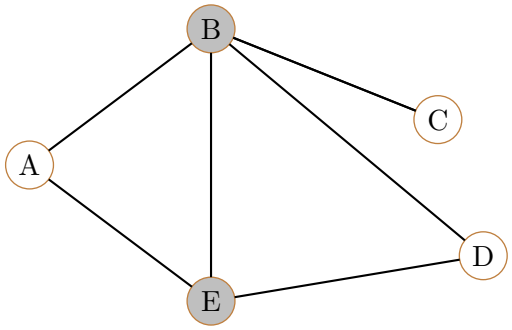

Figure 1.1.: Vertex cover of a graph

Example 1. Consider Figure 1.1, which illustrates a graph and the shaded nodes correspond to the vertex cover of the graph.

If we were to eliminate edges or vertices in a graph, we can observe that the vertex cover obtained by eliminating the same vertices in a given vertex cover will yield a vertex cover for the corresponding subgraph. This feature is important and we shall make implicit use of it, when we consider the toplogical ranking to be defined in Chapter 5 in conjunction with cover ideals which will be introduced in Chapter 6.

1.5. Arithmetic in Finite Fields

A *monomial* in $x_1, x_2, \ldots, x_n$ is a product of the form $\prod_{j=1}^{n} x_j^{\alpha_j}$ where all α_j are non-negative integers. The total degree of a monomial is the sum $\sum_{j=1}^{n} \alpha_j$. A polynomial in variables $x_1, x_2, \ldots, x_n$ with coefficients in the ring R is a finite linear combination of monomials.

Let R be a ring and denote by $R[x_1, \ldots, x_n]$ the ring of polynomials with coefficients in R. Any polynomial $f \in R[x]$ where the degree of f is greater than zero and where f cannot be factored into a product of smaller (non-trivial) polynomials is said to be an irreducible polynomial. In other words if $\deg f > 0$ and f cannot be written as $f = gh, \quad g, h \in R[x]$, and g and h are not constant polynomials, then we say f is irreducible. The irreducible polynomials are very useful for constructing finite field extensions. We need the following

Definition 5. Let $\mathbb{F}$ be a non-trivial commutative ring. If every non-zero element in $\mathbb{F}$ has an inverse, then $\mathbb{F}$ is a *field*.

In the engineering sciences, the most common and used fields, in the sense of Definition 5, are the rationals $\mathbb{Q}$, the real numbers $\mathbb{R}$ and the complex numbers $\mathbb{C}$. Other interesting

fields arise from a prime p when we consider the following set $\mathbb{F}_p = \mathbb{Z}/p\mathbb{Z}$ interpreted as the set $\{0, 1, ..., p-1\}$ together with two compositions, namely addition modulo p and multiplication modulo p. Then $\mathbb{F}_p$ is a field, since it is a commutative group with respect to addition and the set $\mathbb{F}_p^* = \mathbb{F}_p \setminus \{0\}$ is a commutative group with respect to multiplication.

A subfield is a subset of a field, which is itself a field, e.g., $\mathbb{Q}$ is a subfield of $\mathbb{C}$. Among the subfields we have

Definition 6. The *prime subfield* of a field $\mathbb{F}$, is the intersection of all subfields of $\mathbb{F}$.

The prime subfield is the smallest proper subfield of a field. A prime subfield is unique in the following sense:

Theorem 1 ([33])**.** *Every prime subfield is isomorphic to either $\mathbb{Q}$ or $\mathbb{Z}_p$, where p is a prime.*

We are now ready to define what is meant by the *characteristic* of a field.

Definition 7. The *characteristic* of a field is zero if the prime subfield is isomorphic to $\mathbb{Q}$ and it is the prime number p if it is isomorphic to $\mathbb{Z}_p$.

A field $\mathbb{F}'$ is said to be an *extension field* of a field $\mathbb{F}$, denoted $\mathbb{F} \subseteq \mathbb{F}'$, if $\mathbb{F}$ is a subfield of $\mathbb{F}'$. Consider a polynomial $f \in \mathbb{F}[x]$ of degree d. Then f is said to *split* an extension $\mathbb{F} \subseteq \mathbb{F}'$ if and only if f can be factored

$$f = \prod_{i=1}^{d} (x - \alpha_i)$$

into linear factors in $\mathbb{F}'[x]$. Hence f splits in $\mathbb{F}'[x]$ when $\mathbb{F}'[x]$ contains the roots α_i, $i = 1, \ldots, d$.

Definition 8. An extension field $\mathbb{F} \subseteq \mathbb{F}'$ is called a *splitting field* of the polynomial f over $\mathbb{F}$ if f splits $\mathbb{F}'$ but does not split any proper subfield containing $\mathbb{F}$.

We call $\mathbb{F}_p$ a prime field and $\mathbb{F}_p^*$ the multiplicative subgroup of $\mathbb{F}_p$. In general we have

Definition 9. The *finite field* $\mathbb{F}_q$ with $q = p^n$, p is a prime and $n \geq 1$ an integer, is the splitting field of $f(x) = x^q - x$ over $\mathbb{F}_p$. The elements of $\mathbb{F}_q$ are the roots of $f(x)$.

The following theorem is a basic and well-known result of finite fields.

Theorem 2. *The multiplicative group of $\mathbb{F}_q$ is cyclic of order $q - 1$.*

There exists many very different proofs of this theorem and here one of them is presented.

Proof. Let h be the maximal order of the elements in $\mathbb{F}_q^*$. This number divides the order of the abelian group $\mathbb{F}_q^*$ by Lagrange's theorem. So $h|(q-1)$ since the order of $\mathbb{F}_q^*$ is $q-1$. We therefore have that $a^h = 1$ for all $a \in \mathbb{F}_q^*$. Since all elements in $\mathbb{F}_q^*$ are roots in $x^h - 1 \in \mathbb{F}_q[x]$ and the number of roots is at most the degree, it must hold that $q - 1 \leq h$. However we also have that h divides $q-1$ and thus $h \leq q-1$. Therefore, $h = q-1$ and since there exists an element, say g, in $\mathbb{F}_q^*$ of order h, this element must generate all of $\mathbb{F}_q^*$, i.e. $\langle g \rangle = \mathbb{F}_q^*$. □

A finite field can be constructed using irreducible polynomials. Given a prime field $\mathbb{F}_p$, we consider the polynomial ring $\mathbb{F}_p[x]$. A finite field extension can be seen as a set of equivalence classes of polynomials in $\mathbb{F}_p[x]$.

Definition 10. A *finite field* of characteristic p with p^n elements is given by $\mathbb{F}_{p^n} = \mathbb{F}_p[x]/f$, where $f \in \mathbb{F}_p[x]$ is an irreducible polynomial of degree n.

A finite field, when viewed as a field extension, is a quotient ring. Every finite field $\mathbb{F}_{p^n}$ can be viewed as the set of all polynomials of degree less than or equal to $n-1$ with coefficients in $\mathbb{F}_p$.

Example 2. Consider the finite field $\mathbb{F}_{7^3} = \mathbb{F}_7[x]/(x^3+6x^2+4)$. This field contains all polynomials of the form ax^2+bx+c with $a,b,c \in \mathbb{F}_7$, i.e. $ax^2+bx+c \in \mathbb{F}_7[x]$.

Whenever $x^n - a \in \mathbb{F}_p[x]$ is irreducible, we write $\mathbb{F}_{p^n} = \mathbb{F}_p[x]/(x^n-a) = \mathbb{F}_p(\sqrt[n]{a})$. Unfortunately it is not always possible to find an irreducible polynomial on the form $x^n - a \in \mathbb{F}_p[x]$.

Example 3. Consider the finite field $\mathbb{F}_{13^5}$ with 371293 elements. Suppose that there exists an element $a \in \mathbb{F}_{13}$ such that $x^5 = a$. Then $x^5 - a$ must divide $x^{13^5} - x$ with remainder zero. It is a necessary condition that the remainder of this division is zero. The division is time consuming, even on a computer and in the next chapter we shall present a result (Theorem 8) that solves this problem once and for all. In this case there is no such $a \in \mathbb{F}_{13}$.

We will now state some well-known results which guarantees the existence and uniqueness of finite fields. Both theorems can be found in [25].

Theorem 3. *For any prime power p^n, there exists a finite field $\mathbb{F}_{p^n}$.*

The uniqueness (up to isomorphism) is well-known too, and we have

Theorem 4. *Any two finite fields of equal order are isomorphic.*

With these two results, we are now certain, that the chosen representation of a finite field is less important, in the sense that results obtained in a finite field can be transferred to any other finite field of equal size. For computations there is a a very important choice to be made in the representation; we seek to represent two fields of different sizes but with equal characteristic using the same defining irreducible polynomial. This problem is considered in the next section.

1.5.1. Finding Subfields of Finite Fields

We show by example, a method of finding subfields of finite fields. Consider

$$\mathbb{F}_{2^4} = \mathbb{F}_2[x]/(1+x+x^4).$$

It is clear that $\mathbb{F}_{2^2}$ is a subfield of $\mathbb{F}_{2^4}$, however not so clear which elements in $\mathbb{F}_{2^4}$ actually yield the desired subfield. Remember that we need to use the addition and

multiplication from $\mathbb{F}_{2^4}$ and not any of the isomorphic counterparts. It is now used that we know that the prime subfield of $\mathbb{F}_{2^4}$ is the same as for $\mathbb{F}_{2^2}$. So we only need to find a few more elements in $\mathbb{F}_{2^4}$ and we are done. A clever search will yield that

$$P = \{0, 1, x + x^2, 1 + x + x^2\}$$

is the desired set of elements. It is easy to see that P is a subfield of $\mathbb{F}_{2^4}$ of order 4, i.e. $P \cong \mathbb{F}_{2^2}$.

We shall now develop a general algorithm for finding subfields in finite fields. The key point to acknowledge here is that although we easily can find a finite field which is isomorphic to some subfield of $\mathbb{F}_q$, it may be more tedious to find a representation which adheres to the irreducible polynomial used to define $\mathbb{F}_q$. The naive algorithm for finding a subfield of order q' would consider every q'-tuple of elements in $\mathbb{F}_q$ and check the axioms to determine if a given q'-tuple is a finite field. There is a much faster approach to finding the representation of a given subfield $\mathbb{F}_{q'}$ in $\mathbb{F}_q$. We utilize the following theorem which we state without proof, since it follows directly by using the division algorithm on universal polynomials.

Theorem 5. *Let $\mathbb{F}_{q'} \subset \mathbb{F}_q$ be a subfield of the finite field $\mathbb{F}_q$ of order q. An element $a \in \mathbb{F}_q$ belongs to the subfield $\mathbb{F}_{q'}$ if and only if $a^{q'} = a$.*

This means we can settle for an algorithm which runs through every element in $\mathbb{F}_q \setminus \mathbb{F}_p$ and determine if the element raised to the power of q' is the element itself. This approach has the advantage of not relying on the irreducible polynomial used to define $\mathbb{F}_{q'}$. It is probably the most efficient algorithm for finding a given subfield $\mathbb{F}_{q'}$ in $\mathbb{F}_q$ which specifically adheres to the irreducible polynomial in $\mathbb{F}_q$.

Example 4. As an example, consider the finite field

$$\mathbb{F}_{2^6} = \mathbb{F}_2[x]/(1 + x + x^3 + x^4 + x^6).$$

Indeed $\mathbb{F}_4$ is a subfield and the elements

$$S = \{0, 1, 1 + x + x^2 + x^3, x + x^2 + x^3\}$$

form a subfield of order 4 in $\mathbb{F}_{64}$. Obviously we have $\mathbb{F}_4 \cong P \cong S$, where $\mathbb{F}_4$ has the canonical polynomial basis representation, P and S borrows the irreducible polynomials from $\mathbb{F}_{16}$ and $\mathbb{F}_{64}$, respectively.

The above analysis clearly shows that there must exist infinitely many equivalent ways of representing a finite field. For this reason the adjective *canonical*, will be introduced to distinguish between an irreducible polynomial of degree n when considering the finite field $\mathbb{F}_{p^n}$ and all the irreducible polynomials of finite fields for which $\mathbb{F}_{p^n}$ is a subfield. For instance, the irreducible polynomial $1 + x + x^3 + x^4 + x^6$ is canonical with respect to $\mathbb{F}_{2^6}$ but not with respect to $\mathbb{F}_{2^2}$. Note that canonical irreducible polynomials are a class of polynomials.

There are situations where it is beneficial to consider non-canonical ways of representing a finite field. For instance if we want to represent a byte as an element in a finite field, we would use $\mathbb{F}_{2^8}$. Representing two bytes, also called a *word*, as an element in a finite field can be done by considering $\mathbb{F}_{2^{16}}$. Data compression is often concerned with mapping elements of a larger field extension to a smaller subfield.

Chapter 2

Algebraic Varieties over Finite Fields

Algebraic geometry is roughly speaking, the study of algebraic varieties, i.e. the study of the roots of multivariate polynomials, and more generally the study of so-called schemes. While we will omit a discussion of scheme theory, it should be mentioned that everything discussed in this chapter, can safely be formulated in terms from scheme theory, since these theoretical frameworks can be seen as opposites.

We study the connection between algebraic varieties over finite fields and how they relate to the finite field from which the variety is defined over. The viewpoint which will be emphasized here, is to fix the defining polynomial(s) of a variety, and by varying the base field, we see connections between geometry and classical number theory.

2.1. Elementary Diophantine Geometry

The following theorem was proved in [1] during the investigation of the connections between Diophantine geometry and finite fields. Theorems which first appeared or was developed in a separate article will be given a citation immediately after the theorem number. This will indicate that either the result was originally developed in the article in question or that it is taken directly from the cited source. If a result is well-known, we will note that the result is well-known, but refrain from citing another source. Sometimes we include a proof anyway.

Theorem 6 ([1])**.** *Over the finite field $\mathbb{F}_{p^n}$ corresponding to the prime p and the integer $n \geq 1$, the equation*

$$x^{p^k} + y^{p^k} = 1$$

has exactly p^n solutions of ordered pairs (x, y) of elements $x, y \in \mathbb{F}_{p^n}$ for any integer $k \geq 1$.

Proof. Let $k \geq 1$ be an arbitrary integer. By rewriting the binomial coefficient

$$\binom{p^k}{r} = \frac{p^k!}{r!(p^k - r)!}, \quad 1 \leq r \leq p^k - 1,$$

we get

$$p^k! = \binom{p^k}{r} r!(p^k - r)! \, .$$

Making use of the prime factorization theorem, it is easily seen that p divides $\binom{p^k}{r}$. By the binomial formula we get then

$$(x + y)^{p^k} = x^{p^k} + y^{p^k}.$$

Together with the obvious relation

$$(xy)^{p^k} = x^{p^k} y^{p^k},$$

this proves that the power map $x^{p^k} : \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}$ defines an isomorphism of the finite field $\mathbb{F}_{p^n}$ onto itself.

From this follows immediately that

$$(x + y)^{p^k} = 1 \quad \text{if and only if} \quad x + y = 1.$$

Clearly this implies that for every one of the p^n elements $x \in \mathbb{F}_{p^n}$, there exists a unique element $y \in \mathbb{F}_{p^n}$ such that

$$x^{p^k} + y^{p^k} = (x + y)^{p^k} = 1.$$

This proves that the equation $x^{p^k} + y^{p^k} = 1$ has exactly p^n solutions. □

The number of solutions to the circle equation over finite fields of even order can be deduced as a special case of Theorem 6.

Corollary 1 ([1])**.** Over the finite field $\mathbb{F}_{2^n}$ corresponding to the prime 2 and the integer $n \geq 1$, the circle equation

$$x^2 + y^2 = 1$$

has exactly 2^n solutions of ordered pairs (x, y) of elements $x, y \in \mathbb{F}_{2^n}$.

The next result shows that solutions comes in multiples of four and this fact will later be shown to extend to all finite fields of odd characteristic.

Theorem 7 ([1])**.** *Solutions to the circle equation*

$$x^2 + y^2 = 1$$

over the finite field $\mathbb{F}_p$ for p odd comes in multiples of four.

Proof. For any odd prime p, you always have the four solutions $(1, 0)$, $(0, 1)$, $(p - 1, 0)$ and $(0, p - 1)$. Suppose now that $(x, y) = (a, b)$, $1 \leq a, b \leq (p - 1)/2$, is a solution. Then $(a, -b) = (a, p - b)$, $(-a, b) = (p - a, b)$ and $(-a, -b) = (p - a, p - b)$ are also solutions. This completes the proof. □

The following theorem extends and generalize a well-known result of Euler. A special case of this theorem will be used later in this chapter, when we seek the number of solutions to the circle equation over finite fields. The current form of the theorem is a generalization and an extension of a classical result of Leonhard Euler, known as Euler's Criterion. This theorem is joint work with Vagn Lundsgaard Hansen in relation to [1, 2].

Theorem 8 (Extended Euler's Criterion, [2])**.** *Let $q = p^n$ for an odd prime p and an arbitrary integer $n \geq 1$. Suppose the number r is a proper divisor in $q-1$. Then it holds that*

1. *An element $a \in \mathbb{F}_q^*$ is an r^{th} power in $\mathbb{F}_q^*$ if and only if $a^{(q-1)/r} = 1$.*
2. *There are exactly $\frac{q-1}{r}$ different r^{th} powers in $\mathbb{F}_q^*$.*

Proof. We exploit that $\mathbb{F}_q$ is the splitting field for the polynomial $f(x) = x^q - x$ in $\mathbb{F}_p$, and is generated by the q roots of $f(x)$; see [23].

The multiplicative group $\mathbb{F}_q^*$ is a cyclic group of order $q-1$. Choose a generator $\gamma \in \mathbb{F}_q^*$. Then the powers γ^i, $i = 1, \ldots, q-1$, runs through all the elements in $\mathbb{F}_q^*$.

Let r be a proper divisor in $q-1$ and put $k = \frac{q-1}{r}$.

The r^{th} power homomorphism $x^r : \mathbb{F}_q^* \to \mathbb{F}_q^*$ maps γ^i into γ^{ir}, showing that the k elements $a_j = \gamma^{jr}$, $j = 1, \ldots, k$, are exactly the r^{th} powers in $\mathbb{F}_q^*$. This proves part (2) of the theorem.

For $a \in \mathbb{F}_q^*$, consider the polynomial $g(x) = x^r - a$ in $\mathbb{F}_q$.

By polynomial division in $\mathbb{F}_q[x]$ we get

$$f(x) = x^q - x = h(x)(x^r - a) + (a^{(q-1)/r} - 1)x,$$

where

$$h(x) = x^{q-r} + ax^{q-2r} + a^2x^{q-3r} + \cdots + a^{(q-3)/r}x.$$

It follows that $g(x)$ is a divisor in $f(x)$ if and only if $a^{(q-1)/r} = 1$.

Since

$$f(x) = x^q - x = x(x - \gamma^1)(x - \gamma^2)\ldots(x - \gamma^{q-1}),$$

it is clear that $g(x) = x^r - a$ is a divisor in $f(x)$ if and only if it splits completely into linear factors, or in other words, if and only if $a \in \mathbb{F}_q^*$ is the r^{th} power of r roots $\gamma^{i_1}, \ldots, \gamma^{i_r} \in \mathbb{F}_q^*$ of $f(x)$.

Combining information we conclude that $a \in \mathbb{F}_q^*$ is an r^{th} power in $\mathbb{F}_q^*$ if and only if $a^{(q-1)/r} = 1$, thus proving part (1) of the theorem. □

We return to an example from the previous chapter.

Example 5. In Example 3 we studied the finite field $\mathbb{F}_{13^5}$ and asked whether or not there exists an element $a \in \mathbb{F}_{13}$ such that $\mathbb{F}_{13^5} = \mathbb{F}_{13}(\sqrt[5]{a})$. We can now prove that there exists no such element $a \in \mathbb{F}_{13}$. Start by noticing that 5 is not a proper divisor of $13^5 - 1$. By Theorem 8 it follows that $x^5 - a$ is not a divisor of $x^{13^5} - x$. Hence there exists no $a \in \mathbb{F}_{13}$ such that $\mathbb{F}_{13}(\sqrt[5]{a}) = \mathbb{F}_{13}[x]/(x^5 - a)$ is a field.

The reasoning in Example 5 can be generalized by Theorem 8. We have

Corollary 2. If there exists no r'th powers in the prime field $\mathbb{F}_p$, then there exists no r'th powers in any extension of degree k where k is divisible by r.

This corollary can be used to determine if it is possible to find an irreducible polynomial of the form $x^k - a \in \mathbb{F}_p[x]$. Since irreducible polynomials can be used to construct finite fields, having polynomials of this form is very convenient.

Example 6. Consider the finite field extension

$$\mathbb{F}_{13^{12}} = \mathbb{F}_{13}[x]/(x^{12} + x^8 + 5x^7 + 8x^6 + 11x^5 + 3x^4 + x^3 + x^2 + 4x + 2).$$

It would be convenient to find another irreducible polynomial, which is not as long as the one above. Consider

$$\frac{13^{12} - 1}{12} = \frac{23298085122480}{12} = 1941507093540.$$

From this we conclude that there exists 12'th powers in $\mathbb{F}_{13}$. Guessing that $2 \in \mathbb{F}_{13}$ is a 12'th power, we consider $x^{12} - 2 \in \mathbb{F}_{13}[x]$. Provided that $2^{(13^{12}-1)/12} = 1$, which in principle can be verified on a computer (caveat: do not try to do it directly), we get that

$$\mathbb{F}_{13^{12}} = \mathbb{F}_{13}[x]/(x^{12} - 2) = \mathbb{F}_{13}(\sqrt[12]{2}).$$

Remark. It is not practical to compute $2^{(13^{12}-1)/12}$ directly. We can prove that $\mathbb{F}_{13^{12}} = \mathbb{F}_{13}(\sqrt[12]{2})$ by verifying that $2^{60} \equiv 1 \pmod{13}$ and that 60 is a proper divisor of $(13^{12} - 1)/12$, which is easy to do on a computer.

2.2. Polynomial Rings over Finite Fields

Recall that an ideal I of a commutative ring R, is a subset of R, which is a group under addition and for all $x \in I$ and $y \in R$ it holds that $xy \in I$. Since the multiplication in R is defined to be commutative we call I a two-sided ideal or simply an ideal. We will not be concerned with the cases where ideals are not two-sided. Non-commutative algebra, plays a very important part when studying rings of differential operators. Most (if not all) cases we consider in this thesis is modelled using commutative rings. We are content with the following specialization of ideals to polynomial ideals.

Definition 11. A set $I \subset \mathbb{F}[x_1, \ldots, x_n]$ is called an ideal if the following are satisfied

1. $0 \in I$ (Existence of neutral element).
2. $f + g \in I\ , \quad \forall f, g \in I$ (Closure).
3. $f\,h \in I\ , \quad f \in I, \forall h \in \mathbb{F}[x_1, \ldots, x_n]$ (Absorption).

Ideals can be organized in different types. We will sum up a few of these. A *proper* ideal of a polynomial ring is an ideal that is not the whole polynomial ring.

A *prime* ideal P of a commutative ring R is a non-trivial proper ideal with the additional property that for any $x, y \in R : xy \in P \implies x \in P \vee y \in P$. For the finite field $\mathbb{F}_q$ itself it is clear by Lagrange's theorem that the only additive subgroups are ones where the order divides q. Since $q = p^n$ for p a prime and $n \in \mathbb{N}$ it must hold that any additive subgroup of $\mathbb{F}_q$ will be a group of order $q' = p^m$ and $m \leq n$. Therefore finding ideals in rings can be done by first considering the additive subgroups and then determine if any of these are ideals. This is not the most efficient method, but will have to do for now.

A polynomial ideal which consists of monomial generators is called a *monomial ideal*. A monomial $x_1^{\alpha_1} \dots x_n^{\alpha_n} \in \mathbb{F}[x_1, \dots, x_n]$ is said to be *square-free* if $\alpha_i \in \{0, 1\}$ for all i such that $1 \leq i \leq n$.

Definition 12. A monomial ideal $J \in \mathbb{F}[x_1, \dots, x_n]$ is called *square-free* if all generators of J are square-free.

There is an almost trivial connection between subfields and subrings in the corresponding univariate polynomial rings.

Proposition 1. *$\mathbb{F}_{q'}[x]$ is a subring of $\mathbb{F}_q[x]$ if and only if $\mathbb{F}_{q'}$ is a subfield of $\mathbb{F}_q$.*

This can be generalized. The notion of a subfield induces a notion of a subgroup on $\mathrm{GL}_n(\mathbb{F}_q)$, the general linear group with elements in $\mathbb{F}_q$. That is, if $\mathbb{F}_{q'}$ is a subfield of $\mathbb{F}_q$, then $\mathrm{GL}_n(\mathbb{F}_{q'})$ is a subgroup of $\mathrm{GL}_n(\mathbb{F}_q)$.

There is a notion of dimension in polynomial rings, called the Krull dimension. This notion becomes important when reviewing research related to phasor measurement unit placement in Chapter 6.

Definition 13. Let R be a commutative ring with unit. For a polynomial ring $R[x_1, \dots, x_n]$, the *Krull dimension* is defined to be

$$\dim(R) = \sup\{d > 0 \mid \text{there is a chain of prime ideals } P_0 \subset \cdots \subset P_n \text{ in } R\}.$$

For a polynomial ring $\mathbb{F}[x_1, \dots, x_n]$ over a field, we have that $\dim(\mathbb{F}[x_1, \dots, x_n]) = n$.

2.3. Monomial Ordering

Let R denote a commutative ring with unit. Many methods in the theory of polynomials rely on an ordering of terms. One reason for introducing a monomial ordering, is to enable a notion of division of multivariate polynomials. For instance we have ordered terms according to the degree of the terms when using the univariate division algorithm. We need a way to distinguish between monomials, e.g., xy^2 and x^2y have equal total degrees.

Definition 14. A monomial ordering on $R[x_1, x_2, \ldots, x_t]$ is a relation $>$ on the set of monomials x^α, where α is an n-tuple of non-negative integers, such that

(i) $>$ is a total ordering on $\mathbb{Z}^n_{\geq 0}$.

(ii) For $a, b, c \in \mathbb{Z}^n_{\geq 0}$, $a > b \implies a + c > b + c$.

(iii) $>$ is a well-ordering on $\mathbb{Z}^n_{\geq 0}$.

The most common orderings are the degree ordering and the lexicographic ordering. We proceed to define both now.

The lexicographic ordering $>_{lex}$ is defined for every pair $a, b \in \mathbb{Z}^n_{\geq 0}$ by considering the point-wise difference $a - b$ and if the leftmost non-zero entry is positive, we say $a >_{lex} b$. In turn the monomials are likewise considered as $x^a >_{lex} x^b$. See [9] for a concise proof that $>_{lex}$ is a monomial ordering.

Example 7. Consider the polynomial $xy^6 + x^5 \in \mathbb{Q}[x, y]$. Fix the monomial ordering to be $x >_{lex} y$. Then we see that $xy^6 < x^5$.

There are $n!$ different ways to define a lexicographic monomial orderings on a polynomial ring $R[x_1, \ldots, x_n]$ with n variables. Example 7 showed one of the two possible lexicographic orderings on $\mathbb{Q}[x, y]$.

We will primarily use a lexicographic monomial ordering, though it should be noted that there exists many alternatives. One such alternative is the graded lexicographic monomial ordering, where total degrees of monomials take precedence over the lexicographic ordering.

Example 8. Consider the polynomial $xy^6 + x^5 \in \mathbb{Q}[x, y]$ again. Fix the monomial ordering to be $x >_{grlex} y$. Then we see that $xy^6 > x^5$. Precedence is given to the total degree, but if the monomials are equal in total degree, the lexicographic ordering takes over, e.g., $x^3y^2 > x^2y^3$.

2.4. Polynomial Division in $R[x_1, \ldots, x_n]$

We give a very concise introduction to polynomial division of multivariate polynomials and present a well-known algorithm for computing these divisions. Everything introduced in this section, except for the examples, can be found in [9].

Theorem 9 ([9])**.** *Fix a monomial ordering $>$ on the polynomial ring $R[x_1, \ldots, x_n]$ over a commutative ring R with unit. Let $F = (f_1, \ldots, f_s)$ be an ordered s-tuple of polynomials in $R[x_1, \ldots, x_n]$. Then every $f \in R[x_1, \ldots, x_n]$ can be written as*

$$f = a_1 f_1 + \cdots + a_s f_s + r,$$

where $a_i, r \in R[x_1, \ldots, x_n]$, and either $r = 0$ or r is a linear combination, with coefficients in R, of monomials, none of which is divisible by any of $LT(f_1), \ldots, LT(f_s)$.

Just like in the polynomial division theorem for univariate polynomial rings, the r is called the *remainder* of the division. This remainder for multivariate polynomial division is not unique, unlike in the univariate case, and the notion of Gröbner bases is needed in order to adjust this lack of structure, see [9]. We postpone the introduction to Gröbner bases to Chapter 4. The algorithm for computing divisions of multivariate polynomials is the following.

Input: $f_1, \ldots, f_s, f$
Output: $a_1, \ldots, a_s, r$
$p := f$;
$r := 0$;
$a_1 := 0$;
$\vdots$
$a_s := 0$;

while $p \neq 0$ **do**
  $i := 1$;
  $d := false$;
  while $i \leq s \ \wedge \ d = false$ **do**
    if $LT(f_i) \mid p$ **then**
      $a_i := a_i + LT(p)/LT(f_i)$;
      $p := p - f_i\, LT(p)/LT(f_i)$;
      $d := true$;
    $i := i + 1$;
    if $d = true$ **then**
      $r := r + LT(p)$;
      $p := p - LT(p)$;
  end
end

Algorithm 1: Division algorithm taken from [9]

To illustrate how to do this by hand we consider an example.

Example 9. We fix the monomial order to be $x >_{lex} y$ on $\mathbb{F}_{2^n}[x, y]$. Consider the polynomial $f = y^{2^n} x + x^{2^n} y \in \mathbb{F}_{2^n}[x, y]$ and the set of polynomials

$$\mathfrak{F} = \{x + y + 1, y^{2^n} + y, x^{2^n} + y^{2^n} - 1\} \subset \mathbb{F}_{2^n}[x, y].$$

Suppose we want to see if f can be written as a linear combination of polynomials in $\mathfrak{F}$.

Start by dividing $LT(f)$ by $LT(x^{2^n} + y^{2^n} + 1)$ which yields $a_1 := y$. Then subtract $y(x^{2^n} + y^{2^n} + 1)$ from f which gives $y^{2^n+1} + xy^{2^n} + y$. The resulting leading term is xy^{2^n}. Dividing this by the $LT(x + y + 1)$ will give us $a_2 := y^{2^n}$. Again by subtraction we are left with $y^{2^n} + y$ which is the only polynomial in $\mathfrak{F}$ which we did not use. Therefore

$$f = y(x^{2^n} + y^{2^n} + 1) + y^{2^n}(x + y + 1) + y^{2^n} + y.$$

Whenever we divide a polynomial f by a set $\mathfrak{F}$ of polynomials, we write $\overline{f}^{\mathfrak{F}}$ for the remainder.

2.5. Affine Varieties over Finite Fields

The comprehensive introduction in [9] will be used as reference. We assume throughout this section that $\mathbb{F}$ is an arbitrary field, e.g., $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ or $\mathbb{F}_q$. Many results hold for more exotic fields, e.g., the field of rational functions, though they are not considered here.

Let $f_1, \ldots, f_j \in \mathbb{F}[x_1, \ldots, x_n]$ be polynomials in the polynomial ring $\mathbb{F}[x_1, \ldots, x_n]$ over the field $\mathbb{F}$. Consider the set

$$\mathcal{V}(f_1, \ldots, f_j) = \{(a_1, \ldots, a_n) \in \mathbb{F}^n \mid f_i(a_1, \ldots, a_n) = 0 \text{ for all } 1 \leq i \leq j\}.$$

The set $\mathcal{V}$ is called the *affine variety* defined by the polynomials $f_1, \ldots, f_j$. When there is no chance of misleading the reader, we shall say that $\mathcal{V}$ is defined over $\mathbb{F}$, to indicate that the base field is $\mathbb{F}$. We say that the variety is in $\mathbb{F}^n$, because $\mathcal{V} \subset \mathbb{F}^n$, to indicate that polynomials are in n variables.

Definition 15. Let $\mathcal{V} \subset \mathbb{F}^n$ be an affine variety over $\mathbb{F}$. Then we define

$$I(\mathcal{V}) = \{f \in \mathbb{F}[x_1, \ldots, x_n] \mid f(a_1, \ldots, a_n) = 0 \quad \text{for all} \quad (a_1, \ldots, a_n) \in \mathcal{V}\}.$$

The following lemma is important, well-known and easy to prove.

Lemma 1. *The set $I(\mathcal{V})$ is an ideal in $\mathbb{F}[x_1, \ldots, x_n]$.*

Proof. First, we observe that the zero polynomial in $\mathbb{F}[x_1, \ldots, x_n]$ is in $I(\mathcal{V})$. Now consider polynomials $f, g \in I(\mathcal{V})$ and let h be an arbitrary polynomial in $\mathbb{F}[x_1, \ldots, x_n]$. Then for an arbitrary point $a \in \mathcal{V}$, we get $f(a) + g(a) = 0 + 0 = 0$. Furthermore we have $h(a)f(a) = 0$. □

To see one of the major differences between algebraic varieties over algebraically closed fields and varieties over general fields, we need

Definition 16. Let $I \subset \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial ideal. The set

$$\sqrt{I} := \{f \mid f^m \in I \quad \text{for some integer} \quad m \geq 1\},$$

is called the radical of I.

For any ideal $I \in \mathbb{F}[x_1, \ldots, x_n]$, the radical $\sqrt{I}$ is an ideal containing I, see [9]. One important feature of the radical ideals are their connection with the ideals of affine varieties. The connection however, is not shared whenever the field has positive characteristic.

Theorem 10 (Strong Nullstellensatz, [9])**.** *Let $\mathbb{F}$ be an algebraically closed field. If J is an ideal in $\mathbb{F}[x_1, \ldots, x_n]$, then $I(\mathcal{V}(J)) = \sqrt{J}$.*

There is an analogue result for finite fields. This result is well-known and is sometimes referred to as the *strong nullstellensatz in finite fields*. The proof is easy and we state the theorem here without proof.

Theorem 11. *For an arbitrary finite field $\mathbb{F}_q$ and ideal $J \subseteq \mathbb{F}_q[x_1, \ldots, x_n]$ it holds that*

$$I(\mathcal{V}(J)) = J + \langle x_1^q - x_1, \ldots, x_n^q - x_n \rangle.$$

As we shall see, once Gröbner bases are introduced, there are other ways to solve the polynomials over finite fields, which does not require trying every combination of elements. For a computational point of view, the benefits of using more advanced methods are greatest when we consider larger systems of polynomials over finite fields. For now, we are content with exhaustive search.

Example 10. Consider the polynomial $x^2 - y^3 \in \mathbb{F}_{43}[x, y]$. If we want to find the roots of this polynomial, there are 43^2 possibilities to check, e.g., $39^2 - 25^3 \equiv 0 \pmod{43}$.

We end this part of the section with a simple but important example of an affine variety. The affine variety

$$\mathcal{V}(x^2 + y^2 - 1) = \{x, y \in \mathbb{R} \mid x^2 + y^2 = 1\}$$

is easily visualized in the plane $\mathbb{R}^2$ and we say that $\mathcal{V}$ is an affine variety over $\mathbb{R}$.

2.5.1. The Circle Equation over Finite Fields

This section follows ideas first investigated by the author in collaboration with V. L. Hansen in the paper [1]. Theorem 12, which is published in [1], is a result which is related to the number of points on an affine variety over a finite field. The main contribution is a shift of viewpoint from fixing the field and letting the polynomials vary, to fixing the polynomials defining the variety and letting the field (in this case finite fields) vary. The theorem can be seen as a way of calculating the number of points on the affine variety $\mathcal{V}(x^2+y^2-1)$ where the polynomial $x^2+y^2-1 \in \mathbb{F}_q[x, y]$ is an element in the polynomial ring $\mathbb{F}_q[x, y]$.

Theorem 12. *For any finite field $\mathbb{F}_{p^n}$ of characteristic p, the number of solutions to the circle equation*

$$x^2 + y^2 = 1$$

over $\mathbb{F}_{p^n}$ is given by the formula

$$N_{p^n} = p^n - \sin\left(p^n \frac{\pi}{2}\right).$$

Proof. For $p = 2$ the result follows by Corollary 1. Hence it only remains to consider the case for an odd prime p. For convenience put $q = p^n$.

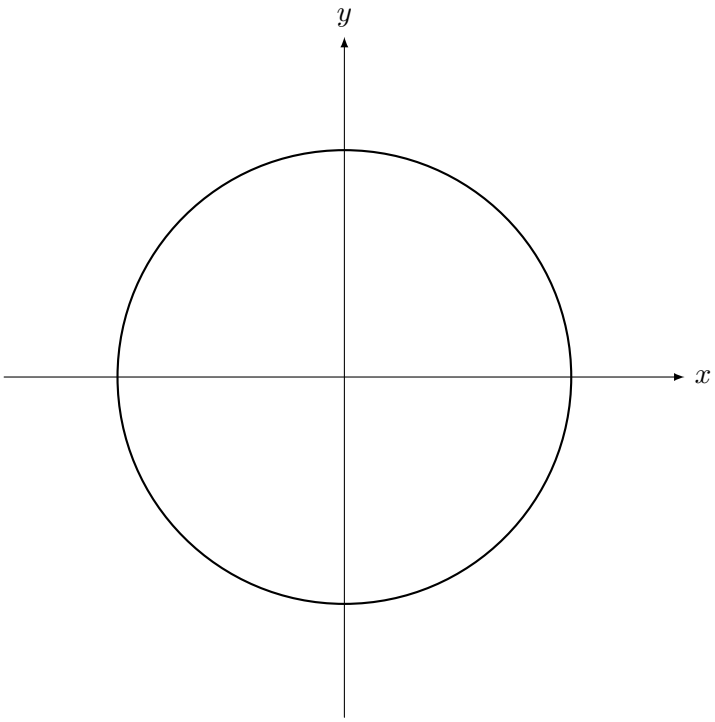

Figure 2.1.: The circle $\mathcal{V}(x^2 + y^2 - 1)$

The multiplicative group $\mathbb{F}_q^*$ is a cyclic group of order $q - 1$, say generated by the element $g \in \mathbb{F}_q^*$, see [23]. Every element in $\mathbb{F}_q^*$ is then uniquely presented as a power g^k of g, where the exponent k is counted modulo q.

We define the multiplicative homomorphism $\eta : \mathbb{F}_q^* \to \mathbb{S}$ of $\mathbb{F}_q^*$ onto the multiplicative group $\mathbb{S} = \{-1, 1\}$, by setting $\eta(c) = (-1)^k$, for $c = g^k \in \mathbb{F}_q^*$.

In the literature η is known as the *quadratic character* of $\mathbb{F}_q^*$. For convenience we set $\eta(0) = 0$.

The squaring homomorphism $x^2 : \mathbb{F}_q^* \to \mathbb{F}_q^*$ maps the element $a = g^l \in \mathbb{F}_q^*$ into $c = g^{2l} \in \mathbb{F}_q^*$. From this we conclude that $c = g^k$ is a square in $\mathbb{F}_q^*$ if and only if k is even modulo q, or equivalently, if and only if $\eta(c) = 1$. Hence there are equally many squares and non-squares in $\mathbb{F}_q^*$. From this follows immediately that

$$\sum_{c \in \mathbb{F}_q} \eta(c) = 0.$$

The number of solutions N_q can be decomposed into a sum of products of the number of solutions $N_q(x^2 = c_1)$ and $N_q(y^2 = c_2)$ to the equations $x^2 = c_1$ and $y^2 = c_2$, for $c_1, c_2 \in \mathbb{F}_q$ with $c_1 + c_2 = 1$. Precisely

$$N_q = \sum_{c_1 + c_2 = 1} N_q(x^2 = c_1) N_q(y^2 = c_2).$$

Observing that the equation $z^2 = c$ over $\mathbb{F}_q^*$ has exactly two solutions if any, the expression for N_q can be rewritten as follows using the quadratic character

$$\begin{aligned} N_q &= \sum_{c_1+c_2=1} [1+\eta(c_1)]\,[1+\eta(c_2)] \\ &= \sum_{c_1+c_2=1} [1+\eta(c_1)+\eta(c_2)+\eta(c_1)\eta(c_2)] \\ &= q + \sum_{c_1\in\mathbb{F}_q} \eta(c_1) + \sum_{c_2\in\mathbb{F}_q} \eta(c_2) + \sum_{c_1+c_2=1} \eta(c_1c_2) \\ &= q + \sum_{c\in\mathbb{F}_q} \eta\left(c(1-c)\right). \end{aligned}$$

Now using that $\eta(4) = \eta(2^2) = 1$ we can further rewrite this as

$$\begin{aligned} N_q &= q + \eta(-1) \sum_{c\in\mathbb{F}_q} \eta(4c^2-4c) \\ &= q + \eta(-1) \sum_{c\in\mathbb{F}_q} \eta\left((2c-1)^2-1\right) \\ &= q + \eta(-1) \sum_{c\in\mathbb{F}_q} \left(-1+\left[1+\eta\left((2c-1)^2-1\right)\right]\right) \\ &= q + \eta(-1)(-q) + \eta(-1) \sum_{c\in\mathbb{F}_q} \left[1+\eta\left((2c-1)^2-1\right)\right]. \end{aligned}$$

By definition of the quadratic character η, the sum

$$S = \sum_{c\in\mathbb{F}_q} \left[1+\eta\left((2c-1)^2-1\right)\right]$$

is the number of solutions in $\mathbb{F}_q$ to the quadratic equation

$$(2c-1)^2 - 1 = a^2,$$

which can be rewritten as

$$(2c-1+a)(2c-1-a) = 1.$$

To solve this product of two linear equations, observe that the factor

$$2c-1+a = \alpha$$

can be chosen arbitrarily in $\mathbb{F}_q^*$. Then necessarily

$$2c-1-a = \alpha^{-1}.$$

By subtraction of equations and division by 2, we get $a = 2^{-1}\left(\alpha - \alpha^{-1}\right)$.

Inserting this value for a into the expression for α yields

$$c = 2^{-1}\left[\alpha + 1 - 2^{-1}\left(\alpha - \alpha^{-1}\right)\right].$$

Since every solution to the quadratic equation in this way turns out to be uniquely determined by a choice of $\alpha \in \mathbb{F}_q^*$ and since the order of $\mathbb{F}_q^*$ is $q-1$, we conclude that the sum S has the value $S = q - 1$.

Collecting facts we get

$$N_q = q + \eta(-1)(-q) + \eta(-1)(q-1) = q - \eta(-1).$$

Now it only remains to determine the value of η on $-1 \in \mathbb{F}_q^*$, i.e. to determine whether -1 is a square, resp. a non-square in $\mathbb{F}_q^*$.

We can choose a generator g of $\mathbb{F}_q^*$ for which $g^0 = 1$, and $g^0, g^1, \ldots, g^{q-2}$ are all the elements in $\mathbb{F}_q^*$, when counting exponents for g modulo $q-1$.

The odd number $q = p^n$ has a unique representation either as $q = 4k+1$ or $q = 4k+3$, for k a non-negative integer.

Suppose $x = g^l$, $1 \leq l \leq (p^n - 1)/2$, is an element with $x^2 = g^{2l} = -1$. Then $g^{4l} = g^{2l}g^{2l} = (-1)(-1) = 1 = g^0$. Consequently

$$4l \equiv 0 \pmod{q-1}.$$

Now suppose $q = 4k+1$. Then we look for solutions to the congruence

$$4l \equiv 0 \pmod{4k}.$$

We get a solution if k divides l, and hence solutions always exist. We conclude that -1 is a square in $\mathbb{F}_q^*$ for $q = 4k+1$.

Next suppose $q = 4k+3$. Then we look for solutions to the congruence

$$4l \equiv 0 \pmod{4k+2}.$$

A solution exists only if $2k+1$ divides $2l$. Since an odd number can never be a factor in an even number, we conclude that the congruence has no solutions and hence that -1 is a non-square in $\mathbb{F}_q^*$ for $q = 4k+3$.

It follows that -1 is a square in $\mathbb{F}_q^*$ if and only if $q \equiv 1 \pmod 4$, and hence

$$\eta(-1) = \begin{cases} 1 & \text{if } q \equiv 1 \pmod 4, \\ -1 & \text{if } q \equiv 3 \pmod 4. \end{cases}$$

In conclusion we get

$$N_q = q - \sin\left(q\frac{\pi}{2}\right),$$

for any prime p, integer $n \in \mathbb{N}$ and $q = p^n$. □

An example of an algebraic variety over a finite field defined by the circle equation can be visualized by points on a lattice. This works well for varieties over prime fields but it needs some interpretation to work over arbitrary finite fields. Figure 2.2 illustrate a way to visualize the varieties over prime fields and the established result that the solutions come in multiples of four, see Theorem 7. The rectangles connect the corresponding quadruples.

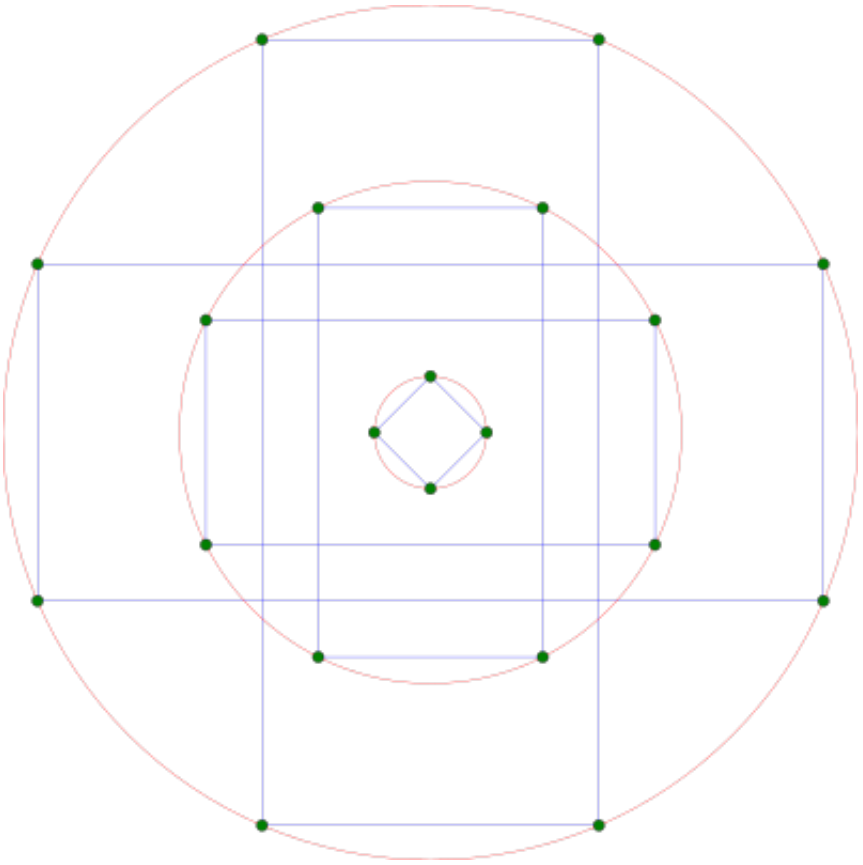

Figure 2.2.: Algebraic variety defined by $x^2 + y^2 - 1 \in \mathbb{F}_{19}[x, y]$.

2.5.2. Patterns in the Number of Solutions to the Circle Equation

We briefly state some consequences to the result concerning the number of solutions to the circle equation over arbitrary finite fields.

Theorem 13 ([1]). *Let p be an odd prime and $n \geq 1$ an arbitrary integer. Then the number of solutions to the circle equation $x^2+y^2 = 1$ over the finite field $\mathbb{F}_{p^n}$ is a multiple of four.*

Proof. The number of solutions is given by

$$N_{p^n} = p^n - \sin\left(p^n \frac{\pi}{2}\right).$$

Since p is an odd prime, $p^n \equiv 1 \pmod 4$ or $p^n \equiv 3 \pmod 4$. On the other hand, clearly

$$\sin\left(p^n \frac{\pi}{2}\right) = \begin{cases} 1, & p^n \equiv 1 \pmod 4, \\ -1, & p^n \equiv 3 \pmod 4. \end{cases}$$

It follows that

$$N_{p^n} \equiv 0 \pmod 4.$$

□

The following theorem settles in which finite fields the circle equation has *diagonal solutions*, i.e. solutions of the form $(x, y) = (x, x)$. Note the connection with the extended Euler's criterion, i.e. Theorem 8.

Theorem 14 ([1])**.** *Let p be an odd prime.*

1. *For an arbitrary integer $n \geq 1$, the circle equation $x^2 + y^2 = 1$ has diagonal solutions over the finite field $\mathbb{F}_{p^n}$ if and only if*

$$2^{(p^n-1)/2} \equiv 1 \pmod p.$$

2. *There are diagonal solutions to the circle equation over the prime field $\mathbb{F}_p$ if and only if $p \equiv \pm 1 \pmod 8$.*

3. *If there are diagonal solutions to the circle equation over a finite field $\mathbb{F}_{p^n}$, then there are exactly two diagonal solutions.*

4. *If there are diagonal solutions to the circle equation over the prime field $\mathbb{F}_p$, then there are also diagonal solutions to the circle equation over $\mathbb{F}_{p^n}$ for all $n \geq 1$.*

Proof. Set $q = p^n$. First suppose that $(x, y) = (a, a)$ is a diagonal solution to the circle equation over the finite field $\mathbb{F}_q$. Then $2a^2 = 1$ and hence $(a^{-1})^2 = 2$, showing that 2 is a square in $\mathbb{F}_q$. Next suppose that 2 is a square in $\mathbb{F}_q$. Then clearly 2^{-1} is also a square in $\mathbb{F}_q$. Therefore there exists an element $a \in \mathbb{F}_q$ such that $a^2 = 2^{-1}$, or equivalently, $a^2 + a^2 = 1$. We conclude that the circle equation has diagonal solutions in the finite field $\mathbb{F}_q$ if and only if 2 is a square in $\mathbb{F}_q$. Notice further that the equation $x^2 = 2^{-1}$ has exactly two solutions $\pm a$, if any.

To finish the proof of the theorem it only remains to determine for which $q = p^n$ the number 2 is a square in $\mathbb{F}_q$. By Theorem 8, it follows that $g(x)$ is a divisor in $f(x)$ and hence that 2 is a square in $\mathbb{F}_q$ if and only if

$$2^{(q-1)/2} \equiv 1 \pmod p.$$

For $n = 1$, i.e. for the prime field $\mathbb{F}_p$, it was known to Gauss (with complete proof) that 2 is a square in $\mathbb{F}_p$ if and only if $p \equiv \pm 1 \pmod 8$, see e.g. Davenport ([10], page 70).

For an arbitrary integer $n \geq 1$, the prime field $\mathbb{F}_p$ is a subfield of $\mathbb{F}_q$. Since the squaring map $x^2 : \mathbb{F}_q^* \to \mathbb{F}_q^*$ is a multiplicative homomorphism mapping $\mathbb{F}_p^*$ into itself, it follows that 2 is a square in $\mathbb{F}_q$ if 2 is a square in $\mathbb{F}_p$.

This completes the proof of the theorem. □

The pattern which we have observed in twin primes modulo 4 makes the following definition interesting.

Definition 17 ([1])**.** A pair of twin primes p and p' for which $p \equiv 3 \pmod 4$ and $p' \equiv 1 \pmod 4$ is called a pair of *siamese twin primes*.

Our main result on the number of solutions to the circle equation in a prime field as a function of the prime can then be given the following concise formulation.

Theorem 15 ([1])**.** *The number of solutions N_p to the circle equation*

$$x^2 + y^2 = 1$$

over $\mathbb{F}_p$ for odd primes, is a strictly increasing function of p in multiples of four, except in pairs of siamese twin primes p and p', where the function stagnates and $N_p = N_{p'}$.

Proof. Let $p < p'$ be a pair of odd prime numbers. It follows by Theorem 12 that $N_{p'} \geq N_p$ and by Theorem 13 that $N_{p'} \equiv N_p \pmod 4$.

Now suppose that $N_{p'} = N_p$. Then necessarily p and p' must be a pair of twin primes.

If $p \equiv 1 \pmod 4$ then $p' \equiv 3 \pmod 4$ since $p' = p + 2$, and hence

$$N_{p'} - N_p = p' - p - \sin\left(p'\frac{\pi}{2}\right) + \sin\left(p\frac{\pi}{2}\right) = 4,$$

which contradicts our assumption that $N_p = N_{p'}$.

On the other hand if $p \equiv 3 \pmod 4$ then $p' \equiv 1 \pmod 4$ and hence

$$N_{p'} - N_p = p' - p - \sin\left(p'\frac{\pi}{2}\right) + \sin\left(p\frac{\pi}{2}\right) = 0.$$

Altogether we conclude that $N_{p'} = N_p$ if and only if p and p' is a pair of siamese twin primes. □

Supported by computer experiments with primes below $n = 2 \cdot 10^9$, there seems to be a curious partition of twin primes into two sets of equal size, namely the siamese twin primes and the non-siamese twin primes. We give

Conjecture 1 ([1])**.** *If we denote by S_n the number of siamese twin primes below n and denote by T_n the number of twin primes below n. Then*

$$\lim_{n\to\infty} \frac{T_n}{S_n} = 2.$$

The conducted computer experiments yield that there are 6388041 pairs of twin primes and 3193559 pairs of siamese twin primes below n.

Chapter 3

Elements of Categorical Algebra

The purpose of category theory was, from its very beginning, to analyze so-called natural transformations. These transformations were the main reason, why S. Eilenberg and S. MacLane defined categories and functors, in other words they wanted to analyze natural transformations, see [11].

Though it began as a theoretical framework in algebraic topology, it has since been established as a research field in its own right. In the context of this thesis, it gives us a way to deal with the modelling of physical systems and transforming them into elegant models which are convenient to work with. This convenience comes from the purely algebraic formulation of structures in our models. It is a powerful tool for modelling systems and the methods introduced here can be considered elementary compared to modern literature.

The subject has become a standard tool in many applied areas of mathematics, e.g., computer science and algebraic systems theory. It is not surprising that the mathematics of structure, i.e. category theory, can be applied in a vast number of seemingly unrelated disciplines.

We follow the modern formulation and notation described in [19] with a few exception, where the foundational text [11] of S. Eilenberg and S. MacLane will be used as reference instead.

3.1. Categories

The strength of categories comes not from the definition of a category but rather from the idea that various structures are related by more than the concrete similarities that might be between any given two objects. In other words, one of the important changes in perspective, from set theory, is the idea that we need to analyze the relationship between objects as well as the objects themselves. And more specifically the relationship between objects is often more important than the objects themselves.

Definition 18. A *category* $\mathcal{C}$ consists of a collection of objects $\mathrm{Obj}\,\mathcal{C}$, a set $\mathrm{Mor}\,\mathcal{C}$ of mappings, also called morphisms, between any two objects and for any object, an identity map, together with an associative composition of maps. A category $\mathcal{C}$ is said to be *small* if $\mathrm{Obj}\,\mathcal{C}$ is a set.

The category consisting of vector spaces over $\mathbb{F}$ as the set of objects and as maps, the linear transformations between vector spaces is called the category of vector spaces. Let $id_{\mathcal{C}}$ denote the identity map in the category $\mathcal{C}$. Similarly we can define the category of rings $\mathcal{K}$, which we define to consist of commutative rings (with unit) as the set of objects and ring homomorphisms as the set of morphisms. In some areas of algebraic systems theory, this category is too narrow as they often study differential rings, e.g., categories with rings of differential operators; these are seldom commutative objects. We define the category having polynomial rings as objects and all maps between these rings as the set of morphisms. This category will be denoted by $\mathcal{P}$ and called the category of polynomial rings.

Definition 19. Every category $\mathcal{C}$ has an *opposite category* $\mathcal{C}^{opp}$ defined with the same objects as $\mathcal{C}$, i.e. $\mathrm{Obj}\,\mathcal{C} = \mathrm{Obj}\,\mathcal{C}^{opp}$, and the arrows in $\mathrm{Mor}\,\mathcal{C}$ are reversed in $\mathrm{Mor}\,\mathcal{C}^{opp}$.

As mentioned in the beginning of Chapter 2, there is a connection between the category of commutative rings and the category of affine schemes. Without going into details of the geometry of schemes, we simply note that the connection is that the category of affine schemes and the category of commutative rings are opposites of each other, see [13].

Practically everything can be considered as a category in the sense that the objects can be the elements of interest and the morphisms can be *all* morphisms between these objects. For instance, electrical circuit diagrams can be taken as the set of objects and we can choose the set of morphisms to be all maps that transform one circuit diagram into another. This seemingly simple way of defining categories has proven to be a useful entrance to mathematics and its applications. In computer science these categories are used to model everything from types to functions, e.g., Haskell[1] types together with Haskell definable functions form a category.

A category which has as objects, the simplicial pairs, and as morphisms all maps between such pairs, is called the category of simplicial pairs. This category is considered in Chapter 5, where it is seen that the objects in this category, induce a ranking on every element in any of the objects.

Several classical viewpoints of graphs in engineering prevail. In [27] among many other studies, networks are analyzed using graphs and one of the features which is considered is the so-called degree of nodes (vertices), which counts the number of edges connected to a node. As we shall see in the Chapter 5, this is far from enough to consider actual network failures etc. When considering graphs, e.g., modeling power system topology, there exists networks where nodes with maximal degree is not the most "critical" nodes in the sense that there exists other nodes of lower degree that yields more damage, when removed. It is necessary to take a more categorical viewpoint on network analysis than what classical graph theory provides.

This suggests that there is much structure that is not analyzed or considered, whenever we model a phenomena by its appearance and not by its relationship to similar objects. The hidden relations between objects of the same kind are of fundamental importance

[1] Haskell Programming Language

in complex networks. Indeed we are interested in relating structures to with other, sometimes similar, structures. This leads us to the next section.

3.2. Functors

Functors are mappings between categories, which are sometimes different. One of the most important (and defining) features of a functor, is that it maps morphisms between objects in such a way that it preserves composition of mappings.

Definition 20. Given categories $\mathcal{C}, \mathcal{D}$, a *covariant functor* is a map $F : \mathcal{C} \to \mathcal{D}$, such that for each object A of $\mathcal{C}$ there is assigned an object $F(A)$ of $\mathcal{D}$ and to each map f in $\mathcal{C}$ there is assigned a map $F(f)$ in $\mathcal{D}$ with the requirement that $F(id_{\mathcal{C}}) = id_{F(A)}$ and $F(f \circ g) = F(f) \circ F(g)$ whenever the composition $f \circ g$ in $\mathcal{C}$ makes sense.

Given the covariant functor $F : \mathcal{C} \to \mathcal{D}$. A *contravariant functor* is the covariant functor $\mathcal{C}^{opp} \to \mathcal{D}$. This trickery of words hides yet another functor, namely the opposite functor ${}^{opp} : \mathcal{C} \to \mathcal{C}^{opp}$. This functor is very important as it gives us a way to describe contravariant functor as covariant functors.

A functor often used in algebraic topology and systems theory is the Hom functor. We define this now. Let A and B be R-modules. The set

$$\mathrm{Hom}_R(A, B) = \{f \mid f : A \to B\}$$

of all R-module homomorphisms f of A into B is an abelian group, under addition, defined for $f, g : A \to B$ by $(f + g)a = fa + ga$. For a fixed left R-module B,

$$\mathrm{Hom}_R(A, B) : {}_R\mathrm{Mod} \to \mathrm{Ab}$$

is a contravariant functor of A. Likewise if A is fixed, then $\mathrm{Hom}_R(A, B)$ is a covariant functor of B. These functors are collectively called Hom functors. The Hom functors are not only defined on ${}_R\mathrm{Mod}$ and Mod_R; trivially the category $\mathrm{Vec}_{\mathbb{F}}$ of vector spaces over the field $\mathbb{F}$ together with linear tranformations shares part of the domain of the Hom functors. A functor $F : \mathcal{C} \to \mathcal{C}$ from a category $\mathcal{C}$ to itself, is called an *endofunctor*.

Among the many important functors, which are often encountered in mathematics are

- The general linear group functor, $\mathrm{GL}_n : \mathcal{K} \to \mathcal{G}$.
- The multiplicative subgroup functor, $(-)^* : \mathcal{K} \to \mathcal{G}$.

A mathematical model is used to relate structures (or concepts). So it is clear that a mathematical model can be represented as a functor. In Chapter 6 we model a problem related to (hyper)graphs via cover ideals which belong to the category of polynomial rings. There are many ways that the process of calculating vertex covers can be represented as a functor, and one such way is to consider the model as a functor from the category of hypergraphs to the category of polynomial rings. It should be noted that the category of graphs can be seen as a subcategory of the category of hypergraphs, yet

there are often used different methods for solving the vertex cover problem depending on whether or not the problem is defined for graphs or hypergraphs. The functorial approach of sending the vertex cover problem into the category of polynomial rings is the same for the two types of categories.

Functors can, like categories, be defined for any kind of application. If we consider the category **Hask** formed by taking Haskell types for objects and Haskell functions between types as morphisms, we can define various functors on this category, e.g., so-called *Maybe* and *List* functors etc. In order to realize this, it may be necessary to define how morphisms are mapped.

Example 11. The *List* functor can be given the following definition.

```
instance Functor List where
  fmap = map
```

Next, we consider the concept of *Maybe* in Haskell. It can be given the following definition.

```
instance Functor Maybe where
  fmap f (Just x) = Just (f x)
  fmap f Nothing = Nothing
```

Notice that in most applications, the domain and target of a functor is implicit. In this cas we work on **Hask**.

Example 12. Let **List** denote the sub-category of **Hask** with lists as objects and with all morphisms taking lists to lists, e.g., "reverse" is a function which takes a list and reverses the order and hence $reverse \in \mathrm{Mor}\,\mathbf{List}$. Consider the mapping $[\,] : \mathbf{Hask}^n \to \mathbf{List}$ from the n-fold Cartesian product of **Hask** to **List** defined by taking n equivalent types (objects) in **Hask** and inserting them into a list in **List**. Then $[\,]$ is a functor.

Remark. There is not a unique way of defining or viewing functors. The list functor $[\,]$ can also be viewed as an endo-functor on the category of sets. The description depends on the purpose of modelling and the setting in which the modelling takes place.

Example 13. Let **List** denote the category of lists as in the previous example. Then a morphism $rmdups : [\,] \to [\,]$ in the category **List** that removes duplicates can be defined in Haskell as

```
rmdups :: (Ord a) => [a] -> [a]
rmdups = map head . group . sort
```

3.3. Natural Transformations

Let $\mathcal{C}$ and $\mathcal{D}$ be two categories. In a very basic form, we consider natural transformations as mappings of functors. More precisely we have

Definition 21. A *natural transformation* $T : F \to G$ between functors $F, G : \mathcal{C} \to \mathcal{D}$, is a map $T_X : F(X) \to G(X)$ for every $X \in \operatorname{Obj}\mathcal{C}$ where the diagram

$$\begin{array}{ccc} F(X) & \xrightarrow{F(f)} & F(Y) \\ {\scriptstyle T_X}\downarrow & & \downarrow{\scriptstyle T_Y} \\ G(X) & \xrightarrow[G(f)]{} & G(Y) \end{array}$$

commutes for all morphisms $f : X \to Y$ in $\operatorname{Mor}\mathcal{C}$ and $X, Y \in \operatorname{Obj}\mathcal{C}$.

Natural transformations can be used as morphisms in a category where the objects are functors.

If a natural transformation $T : F \to G$ between functors $F, G : \mathcal{C} \to \mathcal{D}$ is an isomorphism, then we call T a natural equivalence and write $F \leftrightarrows G$. Further we define the equivalence of two categories $\mathcal{C}$ and $\mathcal{D}$ if there exists two functors $F : \mathcal{C} \to \mathcal{D}$ and $G : \mathcal{D} \to \mathcal{C}$ such that $F \circ G \leftrightarrows id_{\mathcal{D}}$ and $G \circ F \leftrightarrows id_{\mathcal{C}}$ where $id_{\mathcal{C}}$ and $id_{\mathcal{D}}$ are identity functors in $\mathcal{C}$ and $\mathcal{D}$, respectively.

These natural transformations play an important role in homology theory, where they often appear, e.g., relative homology is natural. The determinant is another example of a natural transformation, see [22].

Given a commutative ring K and the group K^* of invertible elements of K, i.e. consider the functor $(-)^* : \mathcal{K} \to \mathcal{G}$ from the category $\mathcal{K}$ of commutative rings to the category $\mathcal{G}$ of groups. Now consider yet another functor, $\mathrm{GL}_n : \mathcal{K} \to \mathcal{G}$, from the category of commutative rings to the category of groups.

The determinant $\det_K : \mathrm{GL}_n \to K^*$ is natural in the sense that the following diagram commutes, whenever $f : K \to K'$ is a morphism of commutative rings, i.e. when $f : K \to K'$ is a ring homomorphism.

$$\begin{array}{ccc} \mathrm{GL}_n K & \xrightarrow{\mathrm{GL}_n f} & \mathrm{GL}_n K' \\ {\scriptstyle \det_K}\downarrow & & \downarrow{\scriptstyle \det_{K'}} \\ K^* & \xrightarrow[f^*]{} & K'^*. \end{array}$$

Example 14. Consider the Frobenius automorphism $\phi : \mathbb{F}_q \to \mathbb{F}_q$, given by $a \mapsto a^p$, where $a \in \mathbb{F}_q$, $q = p^m$, p is a prime and m is a positive integer. One might ask whether it makes a difference to apply the Frobenius automorphism before or after calculating the determinant?

The diagram

$$\begin{array}{ccc} \mathrm{GL}_n\mathbb{F}_q & \xrightarrow{\mathrm{GL}_n\phi} & \mathrm{GL}_n\mathbb{F}_q \\ \Big\downarrow{\scriptstyle \det_{\mathbb{F}_q}} & & \Big\downarrow{\scriptstyle \det_{\mathbb{F}_q}} \\ \mathbb{F}_q^* & \xrightarrow[\phi^*]{} & \mathbb{F}_q^* \end{array}$$

is clearly a commutative diagram, and thus $\det_{\mathbb{F}_q}$ is a natural transformation. Therefore the answer is that it makes no difference whether the Frobenius automorphism is applied before or after calculating the determinant. To illustrate this further, we consider a simple finite field

$$\mathbb{F}_9 = \mathbb{F}_3[x]/(x^2 + 2x + 2).$$

Next we choose a 2×2 matrix A with elements in $\mathbb{F}_9$, e.g.,

$$A = \begin{pmatrix} x+2 & x+1 \\ x & 2x \end{pmatrix}.$$

Obviously $\det_{\mathbb{F}_9}(A) = 2x(x+2) - x(x+1) = 2 + x + 2 = x + 1$. Applying Frobenius to this value yields $\phi(x+1) = 2x + 2$. If we on the other hand consider the Frobenius automorphism applied prior to calculating the determinant, we get the matrix

$$\tilde{A} = \begin{pmatrix} 2x & 2x+2 \\ 2x+1 & x+2 \end{pmatrix},$$

and $\det_{\mathbb{F}_9}(\tilde{A}) = 2 - x = 2x + 2$.

This simple example showed one kind of reasoning that natural transformations can facilitate. In other words, if a map is natural, it will tell us much about the flexibility of such a map with respect to the processes that it is related to. This is a very powerful tool, not just from a mathematical point of view, but also from an engineering one. This form of reasoning is important in designing algorithms, communication protocols etc.

3.3.1. Some Results Related to Determinants

Consider the determinant map $\det : \mathrm{GL}_n \to (-)^*$. As was established in the previous section, this map is natural and we have

$$(\det \mathrm{GL}_n f)A = (f^* \det)A,$$

for every $A \in \mathrm{GL}_n K$, where K is a commutative ring and $f : K \to K$ is a ring homomorphism on K. Furthermore the determinant is a homomorphism, and it holds that for every $k-$fold product of a matrix $A \in \mathrm{GL}_n K$, we get

$$\det(A)^k = \det A^k.$$

Suppose that $k = p$, where p is a prime. Let q be a power of p and consider $\det : \mathrm{GL}_n\mathbb{F}_q \to \mathbb{F}_q^*$ together with the induced map $\mathrm{GL}_n f : \mathrm{GL}_n\mathbb{F}_q \to \mathrm{GL}_n\mathbb{F}_q$, where f is defined by $x \mapsto x^p$, $x \in \mathbb{F}_q$. Let $X \in \mathrm{GL}_n\mathbb{F}_q$ and consider

$$\det(X^p) = \det(X)^p = (f^* \det)X.$$

It follows immediately that we have the following

Proposition 2. *Given a homomorphism $f : \mathbb{F}_q \to \mathbb{F}_q$ of finite fields defined by $x \mapsto x^p$. Then for any $A \in \mathrm{GL}_n\mathbb{F}_q$,*

$$\det(A^p) = \det(\mathrm{GL}_n f A).$$

It is clear that $\det(A)^k = \det A^k$ for all $k \geq 1$, but most often an arbitrary k-fold product of matrices will not yield the same determinant as $(\det \mathrm{GL}_n f)A$, where f is the k'th power map $x \mapsto x^k$.

One can establish how many matrices $\mathrm{GL}_n\mathbb{F}_q$ contains, that is matrices with non-zero determinant. Suppose we want to build an $n \times n$ matrix in $\mathrm{GL}_n\mathbb{F}_q$. The rows needs to be linear independent and non-zero for the determinant to be non-zero. So choose the first row to be any non-zero row; we can do so in $q^n - 1$ different ways. Next choose the second row such that it is linearly independent of the previous row; we can do so in $q^n - q$ different ways since there exists q multiples of the first row. Proceeding with this line of argument will yield the desired result. We have

Theorem 16. *The group $\mathrm{GL}_n\mathbb{F}_q$ has order $(q^n - 1)(q^n - q)(q^n - q^2)\dots(q^n - q^{n-1})$.*

Chapter 4

Gröbner Bases and Their Applications

The theory of Gröbner bases will be introduced. The circle equation is revisited and a few conjectures related to the circle equation are discussed. The relation between graph theoretical concepts and Gröbner bases will be mentioned and we introduce a fast method for computing intersection of ideals, which are relevant in graph theory, the theory of hypergraphs and algebraic geometry. We show empirically that the ordering of a sequence of ideals to be intersected is important. This is illustrated by proposing a ordering algorithm, which we find to yield the fastest intersection of square-free monomial ideals in a sequence. The main advantage of this ordering algorithm, is that we need not develop a new sorting algorithm. Traditional sorting algorithms, like *Bubble sort* or the more efficient *Merge sort* algorithms can be used. The ordering algorithm is invariant with respect to the choice of sorting algorithm.

4.1. S-polynomials

Let $\mathbb{F}[x_1, \ldots, x_n]$ be a polynomial ring over an arbitrary field. For a non-zero polynomial $f \in \mathbb{F}[x_1, \ldots, x_n]$, the *multidegree* of f with respect to the monomial ordering $>$, is given by the maximum of exponents of non-zero terms in f with respect to the monomial order. The multidegree is an n-tuple of non-negative integers. We write $LT(f)$ for the leading term of f and write $LM(f)$ for the corresponding monic (leading coefficient 1) monomial.

Definition 22. For polynomials $f_1, f_2 \in \mathbb{F}[x_1, \ldots, x_n]$, the *S-polynomial* is defined by

$$S(f_1, f_2) = \frac{LCM(LM(f_1), LM(f_2))}{LT(f_1)} f_1 - \frac{LCM(LM(f_1), LM(f_2))}{LT(f_2)} f_2.$$

Example 15. Fix the monomial order to be the lexicographic ordering in which $x < y$.

Consider the ideal $\langle f_1, f_2\rangle = \langle yx^{17} - yx, xy^{17} - xy\rangle$. Then

$$\begin{aligned} S(f_1, f_2) &= \frac{LCM(yx^{17}, xy^{17})}{yx^{17}}(yx^{17} - yx) - \frac{LCM(yx^{17}, xy^{17})}{xy^{17}}(xy^{17} - yx) \\ &= \frac{x^{17}y^{17}}{yx^{17}}(yx^{17} - yx) - \frac{x^{17}y^{17}}{xy^{17}}(xy^{17} - yx) \\ &= y^{16}(yx^{17} - yx) - x^{16}(xy^{17} - yx) \\ &= yx^{17} - xy^{17}. \end{aligned}$$

There is an important feature of S-polynomials, which will serve as a delicate underpinning of most algorithms using these polynomials. Utilizing the definition of polynomial ideals over fields, it is clear that we have

Proposition 3. *Let $I \subset \mathbb{F}[x_1, \ldots, x_n]$ be a polynomial ideal. Then for any pair of polynomials $f, g \in I$, it holds that $S(f, g) \in I$.*

4.2. Gröbner Bases

We introduce the notion of Gröbner bases and show how they can be used to solve problems in mathematics and engineering. The books [9, 32] are our general references.

Consider a polynomial ideal $I = \langle f_1, \ldots, f_t\rangle \subset \mathbb{F}[x_1, \ldots, x_n]$. Define the leading term monomial ideal, denoted by $\langle LT(I)\rangle$, to be the ideal generated by the leading term of every polynomial in I. The ideal $\langle LT(I)\rangle$ has an infinite number of generators and is usually different from the ideal $\langle LT(f_1), \ldots, LT(f_t)\rangle$ defined by the leading terms of the generators of I.

Definition 23. A *Gröbner basis* is defined to be a subset $\{g_1, \ldots, g_i\}$ of an ideal $I \subset \mathbb{F}[x_1, \ldots, x_n]$ for which

$$\langle LT(g_1), \ldots, LT(g_i)\rangle = \langle LT(I)\rangle.$$

Example 16. Fix the monomial order to be the lexicographic ordering $x < y$. Consider the variety $\mathcal{V}(x^2 + y^2 - 1)$ over $\mathbb{F}_{13}$. Suppose that

$$G = \{x^7 + 11x^5 + 2x^3 + 12x, x^5y + 12x^3y + xy, y^2 + x^2 + 12\} \in I(\mathcal{V}),$$

where $I(\mathcal{V}) = \langle x^2 + y^2 - 1, x^{13} - x, y^{13} - y\rangle \subset \mathbb{F}_{13}[x, y]$. For G to be a Gröbner basis, it must hold that for every element $f \in I(\mathcal{V})$, there exists an element $g \in G$ such that $LT(f)$ is divisible by $LT(g)$.

It is clear that there are some disadvantages to this definition since ideals have infinitely many elements. Therefore a criterion is needed to determine if a set of polynomials form a Gröbner basis. This is solved by the following

Theorem 17 (Buchberger's criterion)**.** *Let I be a polynomial ideal in $\mathbb{F}[x_1, \ldots, x_n]$. A basis $G = \{g_1, \ldots, g_i\}$ of I is a Gröbner basis for I if and only if for every pair of distinct polynomials in G, the corresponding S-polynomial is divisible by G.*

For a proof of Theorem 17, see [8] or [9]. Based on this criterion, it is easy to see that the basis for the ideal in Example 15 is in fact a Gröbner basis, since dividing the S-polynomial by the basis will leave a remainder of zero.

Notice the assumption in Buchberger's criterion where the candidate set of polynomials are a basis for the ideal in question. Any algorithm proposed to construct a Gröbner basis must, if using this criterion, assert or ensure that the Gröbner basis is in fact a basis for the original ideal.

4.2.1. Computing Gröbner Bases

Consider an ideal $I \subset \mathbb{F}[x_1, \ldots, x_n]$ in the polynomial ring $\mathbb{F}[x_1, \ldots, x_n]$ over the field $\mathbb{F}$. Choose a monomial ordering $>$ on $\mathbb{F}[x_1, \ldots, x_n]$. A Gröbner basis for I with respect to $>$ is a finite collection of polynomials $G = \{g_1, \ldots, g_i\} \subset I$ where for every non-zero polynomial $f \in I$ the leading term of f is divisible by the leading term of at least one $g \in G$.

The following algorithm will produce a Gröbner basis upon inputting the generators of an ideal. We omit the proof of this here and only present the algorithm itself.

Let $I = \langle f_1, \ldots, f_t \rangle \subset \mathbb{F}[x_1, \ldots, x_n]$ be a non-trivial polynomial ideal in the polynomial ring on n variables over the field $\mathbb{F}$. Then the following algorithm will produce a Gröbner basis for I, see [9].

Input: $F = (f_1, \ldots, f_t)$
Output: Gröbner basis for I, with $F \subset G$
$G := F$;
repeat
  $G' := G$;
  foreach *pair* $\{p, q\}$, $p \neq q$ *in* G' **do**
    $S := \overline{S(p,q)}^{G'}$;
    if $S(p,q) \neq 0$ **then**
      $G = G \cup \{S\}$;
    end
  end
until $G' = G$;

Algorithm 2: Buchberger's algorithm

For our purposes it will suffice to state the above algorithm, though many computer algebra systems will use faster algorithms. We will mention explicitly if another algorithm is used for computing Gröbner bases.

4.3. Reduced Gröbner Bases

Recall that the leading coefficient of a monomial is considered with respect to the multidegree.

Definition 24. Given an ideal I in the polynomial ring $\mathbb{F}[x_1, \ldots, x_n]$ over the field $\mathbb{F}$. Let G be a Gröbner bases for I. Then we say that G is *reduced* if for all generators f in G, no monomial of f lies in the ideal $\langle LT(G - \{f\})\rangle$ and for every f in G the leading coefficient is the multiplicative identity.

For each monomial ordering and a fixed non-trivial polynomial ideal, the reduced Gröbner basis is unique, see [9]. As mentioned in [9], putting matrices into reduced row echelon form in linear algebra is a special case of the uniqueness of reduced Gröbner bases.

Theorem 18. *There exists only finitely many reduced Gröbner bases for a given polynomial ideal $I \subset \mathbb{F}[x_1, \ldots, x_n]$.*

In practice, we can find a reduced Gröbner basis from any Gröbner basis $G = \{g_1, \ldots, g_i\}$, by removing any generator g_j, that reduces to zero via $\langle G - \{g_j\}\rangle$.

4.3.1. The Circle Equation Revisited

We return to the algebraic variety over finite fields defined by the circle equation. The following conjecture seems true but difficult to prove.

Conjecture 2. *Let $I(\mathcal{V}) = \langle x^2 + y^2 - 1, x^q - x, y^q - y\rangle \subset \mathbb{F}_q[x, y]$ be a polynomial ideal and denote by $\{g_i\}$ the reduced Gröbner basis for $I(\mathcal{V})$ with respect to the lexicographic monomial ordering. Then there exists an i such that*

$$LT(g_i) = \begin{cases} y^{(q+1)/2}, & \text{for } q \equiv 1 \pmod 4, \\ y^{(q+3)/2}, & \text{for } q \equiv 3 \pmod 4, \\ y^q, & \text{for } q \equiv 0 \pmod 2. \end{cases}$$

The following example show the conjecture in a special case.

Example 17. Consider $\mathcal{V}(x^2+y^2-1)$ over $\mathbb{F}_{13}$. Then $I(\mathcal{V}) = \langle x^2+y^2-1, x^{13}-x, y^{13}-y\rangle$. The leading monomial ideal of $I(\mathcal{V})$ is then given by

$$\langle LT(I(\mathcal{V}))\rangle = \langle x^2, y^{(13+1)/2}, xy^{(13+1)/2-2}\rangle = \langle x^2, y^7, xy^5\rangle.$$

In the next example we show how the conjecture looks over a specific finite field extension of $\mathbb{F}_2$.

Example 18. Again, we consider $\mathcal{V}(x^2 + y^2 - 1)$, now over $\mathbb{F}_{2^8}$. It holds that $I(\mathcal{V}) = \langle x^2+y^2-1, x^{256}-x, y^{256}-y\rangle$. Trivially we find the Gröbner basis to be $\{x+y+1, y^{256}+y\}$ from which it is clear that $\langle LT(I(\mathcal{V}))\rangle = \langle x, y^{256}\rangle$.

More generally, the conjecture is true for all $q = 2^n$. This is a consequence of the fact that the reduced Gröbner bases follows the general form given in

Proposition 4. *Let $I(\mathcal{V}) = \langle x^2 + y^2 - 1, x^{2^n} - x, y^{2^n} - y\rangle \subset \mathbb{F}_{2^n}[x, y]$ be a polynomial ideal with respect to the lexicographic monomial ordering. The reduced Gröbner basis for $I(\mathcal{V})$ is given by $\{x + y + 1, y^q + y\}$.*

Proof. We observe that $x^2 + y^2 - 1 = x + y + 1$ in $\mathbb{F}_{2^n}[x, y]$. Thus we get

$$I = \langle x^2 + y^2 - 1, x^{2^n} - x, y^{2^n} - y\rangle = \langle x + y - 1, x^{2^n} - x, y^{2^n} - y\rangle.$$

The last ideal is generated by a Gröbner basis, which can be seen by applying Buchberger's criterion. Now consider the monomial ideal $J = \langle x, y^{2^n}\rangle$. Since at least one monomial in $x^{2^n} - x$ is divisible by a generator in J, it can be concluded that

$$\{x + y - 1, x^{2^n} - x, y^{2^n} - y\},$$

is not a reduced Gröbner basis. We will now apply an effective trick. Since any power of any generator of a polynomial ideal is part of that ideal, we can consider this augmented ideal

$$\mathfrak{F} = \langle x + y - 1, x^{2^n} - x, y^{2^n} - y, x^{2^n} + y^{2^n} - 1\rangle,$$

where $\{x+y-1, x^{2^n} - x, y^{2^n} - y, x^{2^n} + y^{2^n} - 1\}$ is a Gröbner basis for I. By polynomial division we immediately get that $\overline{x^{2^n} - x}^{\mathfrak{F}} = 0$. Thus we can remove $x^{2^n} - x$ from the list of generators for I, and since $x^{2^n} + y^{2^n} - 1$ is a power of another generator in I, we get that $I = \langle x + y - 1, y^{2^n} - y\rangle$.

□

Remark. We have now given an alternative proof of Theorem 6 for the case where $p = 2$. What makes the above proof possible is the fact that we are working with an algebraic variety over a field of characteristic 2. This means that we do not get complicated coefficients during the multivariate polynomial division. Another reason for the simplicity is that all coefficients in the original variety have coefficients in the prime subfield of $\mathbb{F}_{2^n}$. Therefore no polynomial division can produce coefficients outside the prime subfield.

Obviously we have

Corollary 3. Finding the reduced Gröbner basis with respect to the lexicographic ordering for $I(\mathcal{V}) = \langle x^2 + y^2 - 1, x^{2^n} - x, y^{2^n} - y\rangle \subset \mathbb{F}_{2^n}[x, y]$ can be done in polynomial time.

The computational complexity of the algorithms used to calculate Gröbner bases, e.g., Buchberger's algorithm, Faugères F4 and F5 algorithms etc. is very difficult to ascertain. The monomial ordering, the number of variables and the number of generators of the ideal in question, all affect the algorithms dramatically. For most engineering purposes, it will suffice to make computational experiments to determine which parameters and approaches one needs to solve a given problem.

Returning to the above conjecture and in an effort to justify its presence here, we start by considering the case $\mathbb{F}_3$, i.e.

$$F_1 = \langle x^2 + y^2 - 1, x^3 - x, y^3 - y\rangle \subset \mathbb{F}_3[x, y].$$

Using Buchberger's algorithm we consider

$$S(x^2 + y^2 - 1, x^3 - x) = y^2 x,$$

which is not divisible by any leading term in F_1. Thus we add y^2x to the list of generators for F_1, i.e.

$$F_2 = \langle x^2 + y^2 - 1, x^3 - x, y^3 - y, y^2x \rangle \subset \mathbb{F}_3[x, y].$$

By definition we now have that $\overline{S(x^2 + y^2 - 1, x^3 - x)}^{F_2} = 0$. More fortunate, we get that $\overline{S(x^2 + y^2 - 1, y^3 - y)}^{F_2} = \overline{x^2y + y^5 - y^3}^{F_2} = 0$. However

$$\overline{S(x^2 + y^2 - 1, y^2x)}^{F_2} = y^4 - y^2.$$

We add this polynomial to the list of generators and get

$$F_3 = \langle x^2 + y^2 - 1, x^3 - x, y^3 - y, y^2x, y^4 - y^2 \rangle \subset \mathbb{F}_3[x, y].$$

Next we consider $S(y^2x, y^3 - y) = xy$. This polynomial cannot be reduced and as usual we add it to the generators. Finally we get a Gröbner basis for F_1, i.e.

$$G := F_1 = \langle x^2 + y^2 - 1, x^3 - x, y^3 - y, y^2x, xy, y^4 - y^2 \rangle \subset \mathbb{F}_3[x, y].$$

The reduced Gröbner basis $\tilde{G}$ is found by removing any basis polynomial which is reduced to zero by the other basis polynomials. For instance, we see that $x(x^2 + y^2 - 1) + y^2x = x^3 - x$ and thus we remove $x^3 - x$ from the list of generators. Doing this exhaustively yields

$$\tilde{G} = \langle x^2 + y^2 - 1, xy, y^3 - y \rangle \subset \mathbb{F}_3[x, y].$$

It is left to the reader to check that this actually constitute a Gröbner basis.

The more general situation is much more difficult and while the author does not believe that it is impossible solve, it is at least a very hard problem. The main issue arises when we let the degree of a polynomial vary. This introduces a difficulty that the division algorithm has trouble dealing with. We now give pointers to its solution in the form of a conjecture that deals with the structure of the solution to the problem.

Conjecture 3. *Let $I(\mathcal{V}) = \langle x^2 + y^2 - 1, x^q - x, y^q - y \rangle \subset \mathbb{F}_q[x, y]$ be a polynomial ideal with respect to the lexicographic monomial ordering. The reduced Gröbner basis for $I(\mathcal{V})$ is considered in two separate versions.*

1. *If $q \equiv 1 \pmod 4$, we have*

$$G = \left\langle x^2 + y^2 - 1, \sum_{n=0}^{(q-1)/4} a_n y^{(q+1)/2-2n}, \sum_{n=0}^{(q-5)/4} b_n x y^{(q+1)/2-2n-2} \right\rangle \subset \mathbb{F}_q[x, y],$$

   where $a_n, b_n \in \mathbb{F}_p$, $a_0 = b_0 = 1$ and $a_{(q-1)/4} = -1$.

2. *If $q \equiv 3 \pmod 4$, we have*

$$G = \left\langle x^2 + y^2 - 1, \sum_{n=0}^{(q+1)/4} c_n\, y^{(q+3)/2-2n}, \sum_{n=0}^{(q-3)/4} d_n x y^{(q+3)/2-2n-2} \right\rangle \subset \mathbb{F}_q[x, y],$$

   where $c_n, d_n \in \mathbb{F}_p$, $c_0 = d_0 = 1$ and $c_{(q+1)/4} = -1$.

Since many problems that can be solved using Gröbner bases are part of the complexity class NP, and some are proven to be NP-complete or possibly EXPSPACE, we can suspect the worst case complexity of computing Gröbner bases to be "exponential". It is still unclear what complexity is required for computing Gröbner bases, because the underlying problem has a major impact on the computation of Gröbner bases, e.g., proposition 4 shows how easy it is to find all Gröbner bases for the ideals of the varieties defined by the circle equation over finite fields of characteristic 2, while the above conjecture is much more difficult to resolve. The decision problems of finding and verifying Gröbner bases of polynomial ideals is heavily dependent on the polynomial ideal in question and the polynomial ring in which the ideal resides.

Conjecture 3 proposes two types of computational problems in order to find a resolution. The first is to find the coefficients for the sums in one of the ideals. The second problem is to decide whether or not it is a Gröbner basis. A resolution of this conjecture in the form of expressions for the coefficients could very well be a major step towards determining the complexity of computing Gröbner bases and maybe towards resolving the unsolved problem P vs. NP. The last of these problems is so important that the Clay Mathematics Institute has made it one of its millennium problems, which awards a million US dollars for a formal proof that settles the problem. Of these seven millennium prize problems only one has been solved (at the time of writing). This is the Poincaré conjecture and it was solved by Perelman in a series of articles posted on arXiv.org, see [28, 30, 29].

To put this discussion into perspective, it can be enlightening to consider an example. It might be easier to find a series of coefficients such that the generators in conjecture 3 is a Gröbner basis, but it should be a Gröbner basis for the variety defined by the circle equation over $\mathbb{F}_q$ of odd characteristic.

Example 19. Consider $I = \langle x^2 + y^2 - 1, x^{25} - x, y^{25} - y \rangle \subset \mathbb{F}_{25}[x, y]$. Since $q \equiv 1 \pmod 4$, we suspect that the reduced Gröbner basis will take the form

$$G = \left\langle x^2 + y^2 - 1, \sum_{n=0}^{6} a_n y^{13-2n}, \sum_{n=0}^{5} b_n x y^{13-2n-2} \right\rangle.$$

By verification on a computer, choosing (for increasing n)

$$\begin{aligned} a_n &= \{1, -1, 1, 0, -1, 1, -1\}, \\ b_n &= \{1, 0, 1, 1, 0, 1\}, \end{aligned}$$

will yield a reduced Gröbner basis of I. There are 9765625 ways of choosing sequences a_n and b_n in this example. If the first coefficients of a_n and b_n and the last coefficient of a_n were not fixed, we would have over a billion possible ways of choosing the sequences.

In general, there are

$$p^{(q-1)/4-1} p^{(q-5)/4} = p^{(2q-6)/4-1},$$

ways of choosing sequences a_n and b_n. Similar observations can be made for sequences c_n and d_n.

The just considered example shows that exhaustive search might be a bad idea and we must hope for a solution which expresses the coefficients in terms of initially known quantities such as q.

Working over the prime fields, we observe that there is a sequence of primes for which exactly one of the coefficients a_n (or c_n) is zero. The first primes of this sequence are $17, 41, 89, 97, 113, 137$.

These observations serve as an interesting starting point for further study.

4.4. Computing Intersection Ideals

The computation of intersections of two or more ideals, can be achieved by using Gröbner bases. The approach is very classical and by using so-called elimination theory in the correct way, we can compute the desired intersections. This section will primarily focus on a couple of key aspects which can be used to speed up the computation for ideals with a large number of generating polynomials. In particularly we are interested in deriving a parallel algorithm for computing these intersections. This will help make these types of computations more accessible to engineering problems.

Definition 25. The *intersection* of two ideals $I, J \in \mathbb{F}[x_1, \ldots, x_n]$ is the set of polynomials in $\mathbb{F}[x_1, \ldots, x_n]$ that belong to both I and J, i.e.

$$I \cap J = \{f \mid f \in I \text{ and } f \in J\}.$$

It is easy to see that $0 \in I \cap J$ for any pair of polynomial ideals I and J. Since both ideals I and J in Definition 25 are (abelian) groups under addition, we see that $I \cap J$ is closed under addition. If a polynomial in $\mathbb{F}[x_1, \ldots, x_n]$ is multiplied with an element of $I \cap J$, then the product belong to $I \cap J$, because every element in $I \cap J$ belong to I and J, both of which are ideals. We have

Proposition 5. *For polynomial ideals $I, J \in \mathbb{F}[x_1, \ldots, x_n]$, the intersection $I \cap J$ of I and J is a polynomial ideal.*

The basic approach for computing the intersection of ideals is to use so-called elimination theory, see [9]. Given two polynomial ideals $I, J \in \mathbb{F}[x_1, \ldots, x_n]$, consider the ideal $L = tI + (1-t)J \in \mathbb{F}[x_1, \ldots, x_n, t]$. Then

$$I \cap J = L \cap \mathbb{F}[x_1, \ldots, x_n].$$

It is a very beautiful and well-known result that if G is a Gröbner basis for L, then $G \cap \mathbb{F}[x_1, \ldots, x_n]$ is a Gröbner basis for $I \cap J$, see [9].

Example 20. Consider two ideals $I, J \subset \mathbb{F}_{13}(\sqrt{8})[x, y]$, given by $I = \langle xy + y, x^2 - y \rangle$ and $J = \langle y^2, x^2 + y^2 \rangle$. We now compute the intersection $I \cap J$. Introduce a new ideal

$$L = tI + (1-t)J \in \mathbb{F}_{13}(\sqrt{8})[x, y, t],$$

more explicitly,

$$L = \langle t(xy + y), t(x^2 - y), y^2 - ty^2, x^2 + y^2 - x^2t - y^2t \rangle.$$

A Gröbner basis for L is given by $\{x^2 - yt, xyt + yt, y^2 - yt, yt^2 - yt\}$. Denote by G the ideal generated by the Gröbner basis for L, just mentioned. Then

$$G \cap \mathbb{F}_{13}(\sqrt{8})[x, y] = \langle y^3 - y^2, xy^2 + y^2, x^2 - y^2 \rangle.$$

This ideal is the intersection of I and J.

4.4.1. Parallelized Intersection of Ideals

Let k be a commutative ring with unit. A polynomial ideal is by definition an additive group. The intersection $I \cap J$ of two polynomial ideals $I, J \in k[x_1, \ldots, x_n]$ is another polynomial ideal, see Proposition 5, hence it is the group containing all elements which are both in I and J. From this, it is clear that $I \cap J$ is associative under the addition of ideals.

Now we assume that we have four ideals $I_1, I_2, I_3, I_4 \subset \mathbb{F}_q[x_1, \ldots, x_n]$ and we want to compute the intersection of these. Denote by $I_{i,j}$ the intersection of the ideals I_i and I_j. In this case it holds that

$$I_{1,2} \cap I_{3,4} = I_{1,3} \cap I_{2,4} = I_{1,4} \cap I_{2,3}.$$

More generally it is clear that for any number of ideals in $\mathbb{F}[x_1, \ldots, x_n]$ we get that the ordering of intersection is insignificant with respect to the final result. There will, however, be a small delay in performance if one seeks to fully parallelize this task, since results needs to be joined at a certain point. As we shall see later, a number of obstacles needs to be passed in order to get better results from a parallel computation of the intersection of a sequence of polynomial ideals. The main obstacle that we shall find a way around is to determine a proper partial ordering of the intersections. This is done by introducing a natural sorting algorithm.

In practice, parallelizing the intersection of polynomial ideals is a very difficult task. While most libraries can handle accumulative intersections of ideals, many of the freely available libraries are not thread safe, which is needed for computing ideals in parallel.

Another issue may arise, since we do not know if computing $I_{1,4} \cap I_{2,3}$ will take the same amount of time as computing $I_{1,2} \cap I_{3,4}$ or $I_{1,3} \cap I_{2,4}$. These issues are most easily settled by implementing a parallel algorithm and then simulate the situation.

The quite simple definition of the intersection of two ideals, paves the way for the natural parallelization that can be done with respect to computing these intersections. While it is very easy to theoretically observe that this parallelization can be done, it is not always so easy to actually implement it on a computer. Many computer algebra systems exist, that can calculate intersections of polynomial ideals. Most of these are suitable for toying around with, but they all seem to lack one very important feature; scaling computations to a massive amount. While some deliver C++ or Python interfaces, they tend to be optimized for computing a small amount of possibly complicated operations.

We need a system for feeding a very large amount of information to one of these algebra systems. The quickest but not necessarily the best way to do this, was to develop a C++ application that can generate Sage[1] scripts on demand. It is possible that there are more enterprise ready ways of doing this if one uses commercial software. Otherwise one will have to develop all the frameworks from scratch. We focus on special square-free monomial ideals called cover ideals.

Definition 26. The *cover ideal* $J \subset \mathbb{F}[x_1, \ldots, x_n]$ of a graph G with vertex set $V = \{v_1, \ldots, v_n\}$ and edge set $E = \{e_1, \ldots, e_t\}$, is given by

$$J_G = \bigcap_{e \in E} \langle x \mid x \in e \rangle.$$

We develop an algorithm which is not necessarily fully optimized but yields very promising results. All calculations are done in Sage, however we generate the Sage/Python scripts in order to make calculations in polynomial rings with a very large number of variables. The scripts are generated from basic C++ code which can be found in the appendix, along with descriptions for building the software.

The algorithm that the author has developed is very concise. It works for any number of ideals and not only monomial ideals, though performance is only measured on square-free monomial ideals due to their relationship with (hyper)graph theory.

It is assumed that the variable *ideals* is a list of ideals in the data type of *ideal* in Sage. We start by introducing a small function which we map our subdivided ideals onto.

```
def idealIntersection(A):
    return A[0].intersection(A[1])
```

Naturally we need a way to map ideals from a list of small ideals into a set of partitioned ideals. Here we group them two-and-two.

```
def mapIdeals(ideals):
    ideals_par = []
    for t in range(0,len(ideals)-1):
        if (len(ideals)>=2):
            I = ideals.pop()
            J = ideals.pop()
            ideals_par.append([I,J])
    return ideals_par
```

The last thing we need is to consider the overall algorithm, which will continually map ideals and reduce via intersection until there is only a single ideal left.

```
if __name__ == '__main__':
    while ( len(ideals)!=1 ):
        p = mp.Pool()
        results = []
        ideals_par = mapIdeals(ideals)
        p.map_async(idealIntersection, ideals_par, \\
                    callback=results.append)
```

[1] http://www.sagemath.org

```
        p.close()
        p.join()
        for I in results[0]:
            ideals.append(I)

coverIdeal = ideals[0]
print coverIdeal
```

The following results will illustrate the performance of the algorithm. This is done using a quad core Intel Xeon E3 1225, with 24GB memory running Fedora Linux 23, with kernel version 4.4.4-301. The compiler used is GCC 5.3.1. The SageMath version used here is 7.0, and Sage was used to compute the intersection of ideals. Sage uses Python version 2.7.10. As the figures will show, the algorithm more than compensates for the overhead of passing calculations to Sage. This overhead was minimized by doing parallelization in the Sage built-in Python, rather than doing this in C++.

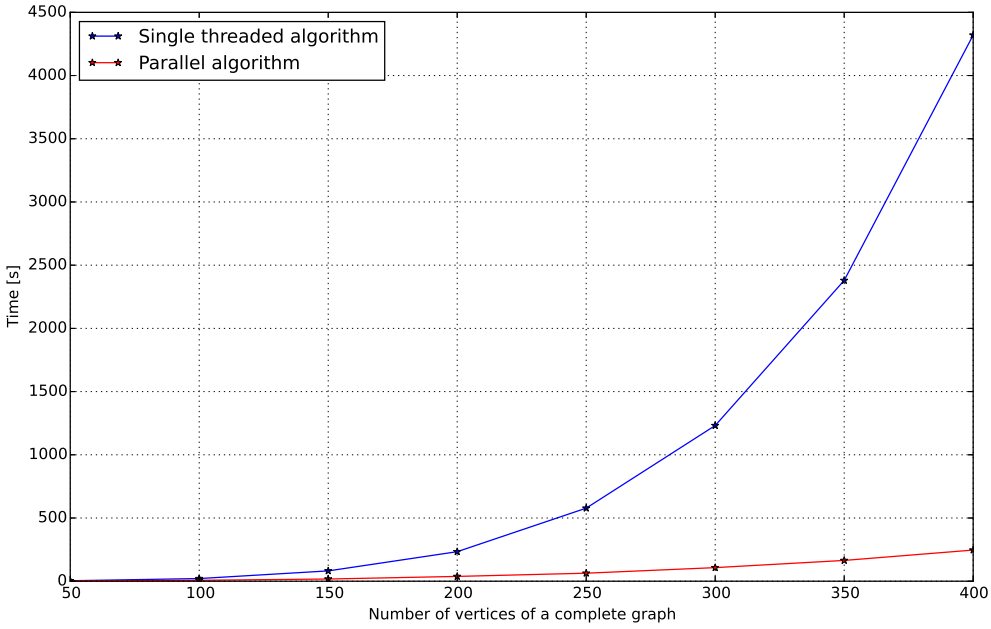

Figure 4.1.: Comparison of algorithms for computation of cover ideals

Figure 4.1 clearly shows the advantages of using a parallel algorithm for computing the intersection of ideals. In terms of speed we get the speedup that we could hope for. How well this parallel algorithm scales on CPU's with alot more cores is unknown at this point. However we can say something meaningful about the memory usage of the current implementations.

Based on the plot in figure 4.2, we see how the parallel implementation scales with the number of threads. In our test setup, we used a quad core Xeon processor and we

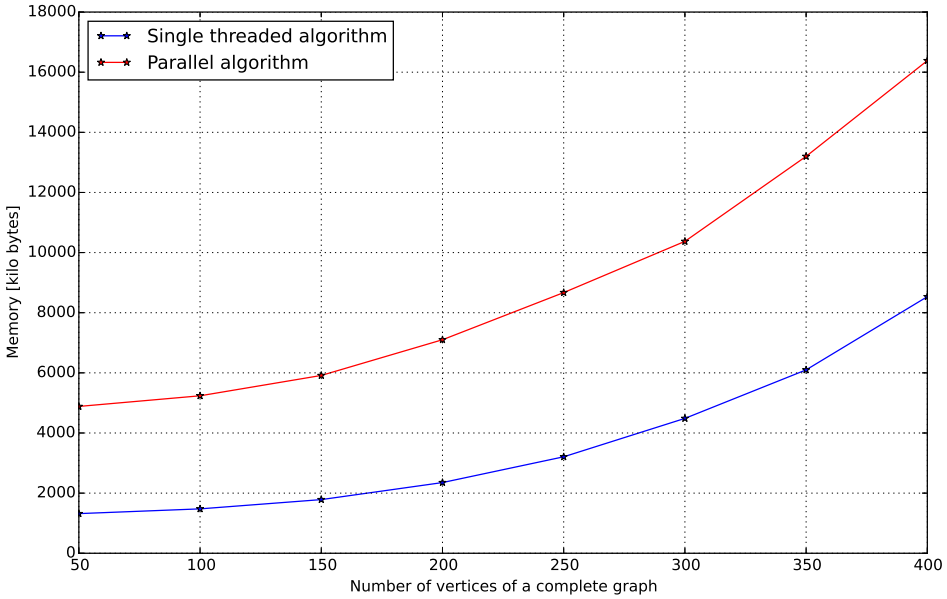

Figure 4.2.: Comparison of memory usage for computation of cover ideals

see that memory usage is roughly 4 times that of the single threaded algorithm. We do see an accumulated memory build-up, meaning that the parallel algorithm uses memory at a slightly faster rate than the single threaded algorithm. Fortunately this problem is minor in our setting.

The real issue arise when we consider random graphs or more specifically, when the random graphs do not provide a structure that makes individual intersection calculations equal. This means that the parallel algorithm would eventually have a worse running time than the single threaded version. We will discuss the reasons for this in the following.

First of all, we must consider if the intersection of a large polynomial ideal with a small polynomial ideal is easier to calculate than that of two medium sized ideals. Secondly it would be interesting to know if the order of the intersection of ideals is of significant importance.

Our initial results are that the order by which the intersections are calculated have a huge impact on the calculation time, no matter which of the algorithms are used, i.e. parallel or single threaded. This was observed because the parallel algorithm performed poorly on random graphs, prompting the author to investigate the reason for this. As a baseline we do 100 calculations each of which is exactly the same. The variation observed is introduced by the operating system of the machine. Figure 4.3 will show the results.

Figure 4.4 shows the same calculation on the same random graph, except the order of the intersection is shuffled at each iteration.

There two things that are important of observe from figures 4.3 and 4.4. The first is

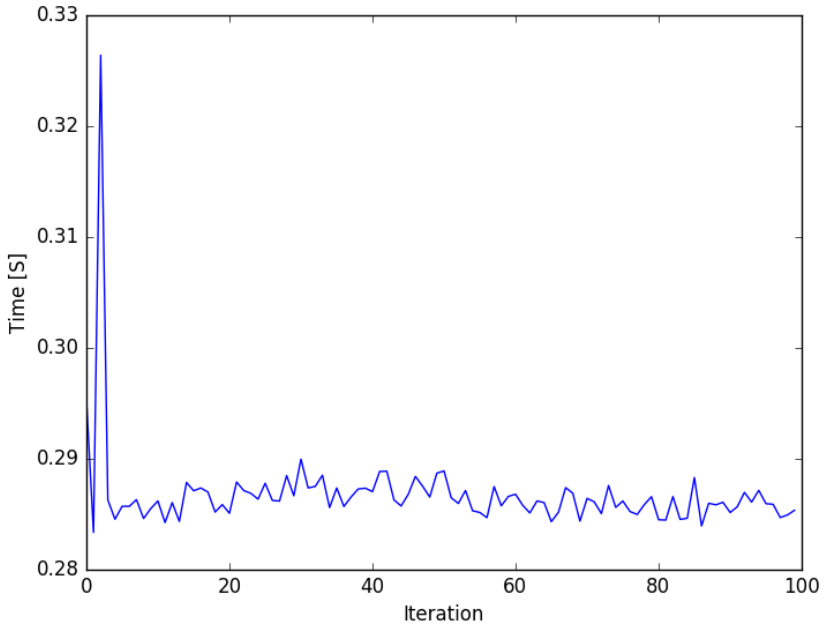

Figure 4.3.: Random graph with intersection order not shuffled, i.e. ordered

that the ordering used in the calculations for Figure 4.3 is much faster than those used in almost all shuffles in the calculations for Figure 4.4. The second important observation is that the variance of the times in Figure 4.4 is very high and comparing with Figure 4.3 it is clear that this variance is a consequence of the order of intersection. So a natural question is: *What is the order of intersection in the baseline example?*

To answer this question we introduce an algorithm, which orders a sequence of ideals in such a way that taking successive intersection is fast. Sorting will take place at two different levels. First, generators of each ideal will be sorted according to the monomial ordering. Secondly, each ideal is sorted according to the first generator, then the second generator etc. again using the monomial ordering as comparator. The following code illustrates a concise implementation of this algorithm and it is optimized for computing cover ideals of random graphs. The lambda function expresses that we sort with respect to first generator and then with respect to the second generator. The lexicographic monomial ordering is a natural part of most modern programming languages and it is utilized in this case. The sorting algorithm utilized in Python's *sorted* function is called *Timsort*, which is a hybrid of the *Merge sort* and *Insertion sort* algorithms, and it has a worst-case running time of $O(n \log n)$. Indeed it may be advantageous to sort a list of ideals in parallel, but as we have established, both the single threaded algorithm and the parallel version suffers under random ordering of intersections. Since both algorithms require this sorting to take place, we omit a discussion of parallelizing sorting algorithms. If, one were to compare this parallel algorithm with other well-known algorithms for

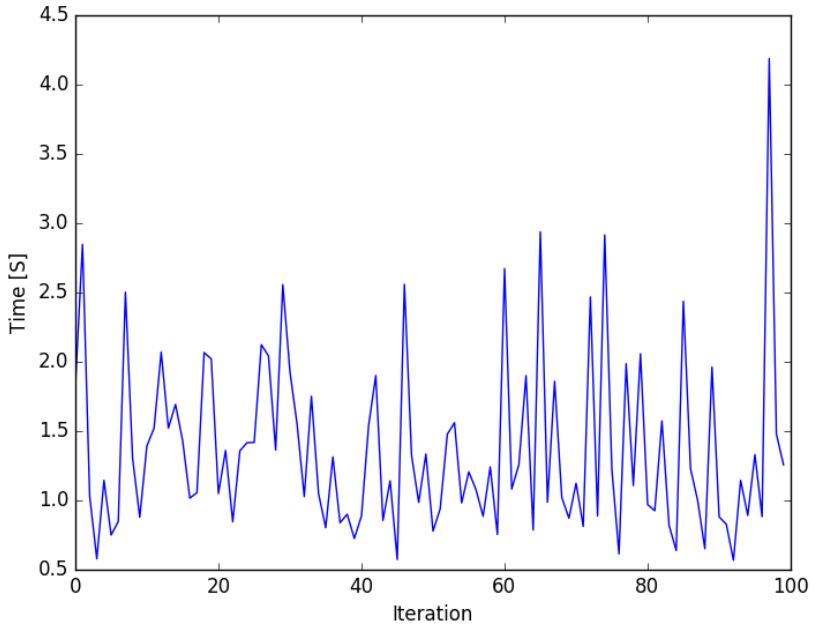

Figure 4.4.: The same random graph with intersection order shuffled

computing vertex covering of (hyper)graphs, then a parallel sorting algorithm should be considered; a parallel version of *Merge sort* is known to exist.

```
for i,I in enumerate(ideals):
    I = ideal(sorted(I.gens(), reverse=True))
    ideals[i] = I

ideals = sorted(ideals,key=lambda I: (str(I.gens()[0]).lower(),\\
    str(I.gens()[1]).lower()))
```

Remark. It was a fortunate event that turned the author's attention to the connection between the order of intersection and the overall computation time. First of all the above ordering is natural when generating Python scripts, where *for* loops often lead to generating the ideals in an sequence, adhering to the lexicographic ordering of monomials. Random graphs were generated by randomly removing edges of a complete graph. This approach often yields graphs with resulting monomial ideals in a random ordering.

As a beneficial consequence of this sorting algorithm, doing intersections in parallel provides advantageous results. It is now investigated if there are improvements to be made if we divide the sequence of ideals into a finite number, e.g., the number of hardware threads, and then compute each collection separately on each thread. Of course the remaining number of ideals from each hardware thread shall be intersected in the end.

The parallel algorithm is updated in the following sense. Instead of mapping the list of sorted ideals two-and-two and passing them to a new thread, we divide the list into a

number of smaller lists, namely the number of cores on the CPU, which in our case is 4 sets.

```
def mappingIdeals(ideals):
    idealer = []
    N = len(ideals)
    for w in range(0,4):
        idealer_tmp = []
        for t in range(0,(N)/4):
            if (len(ideals)>0):
                idealer_tmp.append(ideals.pop())
        if (idealer_tmp != None):
            idealer.append(idealer_tmp)
    return idealer
```

Before we can distribute the mapped lists to each of the cores on the CPU, we need to update the function for computing the actual intersection of ideals, such that it can take a list of many ideals.

```
def idealsIntersection(A):
    cover = A.pop()
    for a in A:
        cover = cover.intersection(a)
    return cover
```

We then pass each of these 4 sets to a separate core and collect the 4 resulting ideals which we then compute in a linear fashion.

```
if __name__ == '__main__':
    p = mp.Pool()
    results = []
    ideals_pairs = mappingIdeals(ideals)
    p.map_async(idealsIntersection, ideals_pairs, callback=results.append)
    p.close()
    p.join()
    coverideal = results[0].pop()
    for I in results[0]:
        coverideal = coverideal.intersection(I)
    print coverideal
```

Contrary to the original hypothesis, this turned out to be worse, when working with random graphs. This can be seen by considering Figure 4.5.

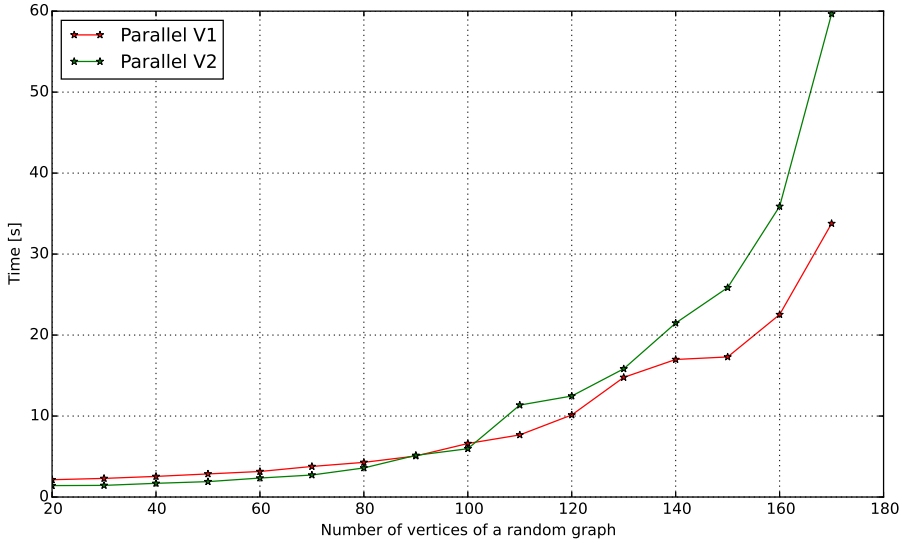

Figure 4.5.: Comparison of parallel algorithms for random graphs

Partitioning the sequence of ideals into a fixed number of subsequences and processing each subsequence separately on a hardware thread seems to be less efficient than computing intersections more refined as the initial version of the parallel algorithm suggests.

All three algorithms, i.e. the single threaded version and the two parallel versions are compared in Figure 4.6.

We end the section by noting, that it seems to be faster to continually map two ideals to a thread for computing intersections than to bulk partition the sequence of ideals into a fixed number of threads. This viewpoint is supported by Figure 4.5. Although we expected a performance increase, when doing intersections in parallel, it was surprising that the ordering was so important, see Figures 4.4, 4.3. Another surprising outcome, was the performance increase gained by doing computations in parallel, see Figure 4.6. In conclusion, the results obtained by doing intersections in parallel have exceeded the authors expectations.

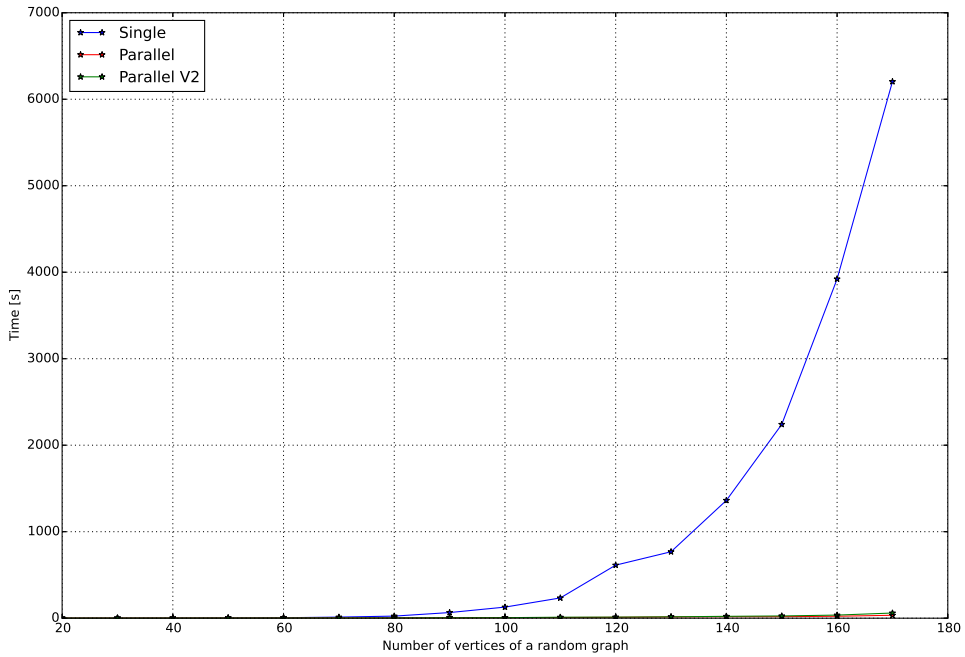

Figure 4.6.: Comparison of all algorithms for random graphs

4.5. Future Research: Applications to Packing Problems

An interesting area of research which was not considered here, is the so-called packing problems and more specifically the *bin packing problem*. The basic setting is to consider a finite space consisting of so-called bins all which are all equal in size. We want to put a set of objects into these bins such that the least number of bins are utilized. This can be demonstrated via a way to model these problems using polynomial ideals and then illustrate the relationship between solutions and normal forms.

The bins are defined as an interval $B = [0; r] \subset \mathbb{R}$. Suppose we have a finite collection of objects $C = \{x \in \mathbb{R} \mid x \leq r\}$. We seek the minimum number of bins needed to contain every element in our collection C. Consider the following naive algorithm for modelling the problem as a polynomial ideal.

Input: B and C
Output: Polynomial ideal model of packing problem
$P :=$ The power set of elements in C.
$I :=$ ideal generated by 0 in $\mathbb{F}[x_1, \ldots, x_n, w]$.
foreach p *in* P **do**
  Denote by $p_i \in C$ the i'th element in p.
  if $\sum p_i \leq r$ **then**
    $I := \langle (\prod p_i) - w \rangle + I$.
  end
end
return I

Algorithm 3: Bin model algorithm

Now imagine that we are interested in determining the minimal number of bins which can contain all objects in C. Consider the polynomial ideal I generated by the algorithm above. Use the graded lexicographic ordering $<_{grlex}$. Then the normal form of any feasible solution will have multidegree that is minimal among all equivalent reductions. Further such a normal form is unique, see [9].

4.5.1. Example of Bin Packing

In 2004, Joseph Malkevitch wrote a series of articles for the American Mathematical Societies feature column. In this series, Malkevitch describes the problem of 1-dimensional bin packing. The example used in one of these articles was as follows: *Given bins of size 10. How few are needed to store weights* $\{3, 6, 2, 1, 5, 7, 2, 4, 1, 9\}$*?*

Let w correspond to a bin and $p = \{3, 6, 2, 1, 5, 7, 2, 4, 1, 9\}$. Let us denote by a the smallest object in the sequence p and by h the largest number in p.

The power set contains 1024 sets and upon removal of trivial duplicates only 976 remains. Only 95 (including the empty set) sets remain once the restriction of the bin capacity is taken into account. We can however make do with far less. In fact it is

sufficient to consider the following polynomial ideal.

$$I = \langle a^{10} - w, b^5 - w, ac^3 - w, bd^2 - w, e^2 - w, df - w, cg - w, ah - w, ab^2 - e \rangle.$$

Now calculate the normal form of the polynomial $F = a^2b^2cdefgh$, which yields w^4 showing that no less than 4 bins are needed to pack all objects/weights. This computation is easily done on a computer, however not so by hand. The Gröbner basis of I consists of 54 generating polynomials, showing that doing these calculations by hand is not a trivial task.

Chapter 5

Applied Homology Theory

The most elementary notion which shall underline most analysis in this chapter is that of connectedness. A homology theory is a sequence of functors H_n which measures (among other things) the shape, such as connectedness, of the underlying objects. There exists many homology theories and in this text, we follow the homology functors from the category of simplicial pairs to the category of abelian groups. More specifically we define simplicial homology to be the relative homology of simplicial complexes. We follow [31] very closely and draw upon formulations made in [18].

We begin by introducing the categories that we will be working with. Of these categories, the most neglected but often most useful category is the category with a single object and all possible morphisms between this object. This type of category will be used to model engineering objects and relate their inherent topological strengths and weakness in the form of a ranking which we will introduce later in this chapter. The results were first published in [3] and later in a refined version in [4].

5.1. The Category of Simplicial Pairs

Simplicial complexes are very important in computational topology and they are one of the core objects used in computational homology theories. We opt for a combinatorial definition, which is slightly more general than geometric simplicial complexes. The combinatorial version is called abstract simplicial complexes. We give

Definition 27. Let D be a discrete set. An *abstract simplicial complex* with 0-simplices from D is a collection X of finite subsets of D, such that for each $\sigma \in X$ all subsets of σ are also in X. A subset $\sigma \in X$ with $k+1$ elements is called a k-simplex.

A *subcomplex* of an abstract simplicial complex X is an abstract simplicial complex A such that every simplex in A is a simplex in X. A pair of abstract simplicial complexes is understood as a pair (X, A) where X is an abstract simplicial complex and A is a subcomplex of X. A pair of abstract simplicial complexes will be termed an *abstract simplicial pair*. Abstract simplicial pairs form a category with the abstract simplicial pairs as objects and all set maps between such pairs as morphisms.

5.2. The Category of Chain Complexes

Most homology theories factor through the category of chain complexes. This is in fact what we intend to do. Therefore we define this important category.

Definition 28. A *chain complex* is a sequence of abelian groups, indexed by the integers, connected by homomorphisms such that the composition of two such morphisms is zero.

A chain complex, when seen as an object, can be transformed into another (or the same) chain complex. A map transforming one chain complex into another (or the same) chain complex is called a chain map. More precisely we have

Definition 29. A *chain map* is a homomorphism between chain complexes such that the boundary operators commute, i.e. a chain map is a collection of homomorphisms $\{\tau : C_q \to {C'}_q\}$ such that each square

$$\begin{array}{ccc} C_q & \xrightarrow{\partial_q} & C_{q-1} \\ \downarrow{\scriptstyle \tau} & & \downarrow{\scriptstyle \tau} \\ {C'}_q & \xrightarrow{\partial'_q} & {C'}_{q-1}. \end{array}$$

commutes.

Like simplicial pairs, chain complexes form a category with chain complexes as objects and chain maps as morphisms.

For instance, consider the following sequences of commutative groups

$$0 \longrightarrow \mathbb{F}_{2^8} \longrightarrow \mathbb{F}_{2^4} \longrightarrow 0,$$

and

$$0 \longrightarrow \mathbb{Z} \times \mathbb{Z} \longrightarrow \mathbb{Z} \longrightarrow \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z} \longrightarrow \mathbb{Z} \times \mathbb{Z} \longrightarrow 0.$$

Whether these sequences constitutes chain complexes, depends on the morphisms. However as long as the composition of two (consecutive) morphisms is zero, they belong to the category of chain complexes.

Consider a chain complex of abelian groups A_k,

$$\ldots \longrightarrow A_{k+1} \xrightarrow{\partial_{k+1}} A_k \xrightarrow{\partial_k} A_{k-1} \longrightarrow \ldots$$

Definition 30. The k'th *homology group* is given by

$$\mathrm{H}_k(A_k) = \ker \partial_k / \mathrm{im}\, \partial_{k+1}.$$

The operator $\partial_k : A_k \to A_{k-1}$ is called the *boundary operator*. By definition $\partial\partial = 0$, and so whatever boundary operator one desires to use, one must ensure that this assumption is true. Since the boundary of a boundary is zero, it is clear that $\mathrm{im}\, \partial_{k+1} \subset \ker \partial_k$ for all $k \in \mathbb{Z}$.

5.3. Relative Homology

We note that relative homology of chain complexes can be defined whenever the chain groups are objects in a suitably chosen category. These categories were introduced in its correct form by Grothendieck [16] and are called abelian categories, see [21] for an explanation. It is sufficient for our purposes here, to note that the category of vector spaces over a fixed field is an abelian category.

Relative homology in the category of simplicial pairs will be explained. Since a simplicial pair consists of a simplicial complex and a subcomplex, the chain groups needs to take this relationship into account. The most natural way of doing this is nicely illustrated in [31], by introducing the notion of relative chain groups. Define the relative chain group of a pair (X, A) as

$$C_q(X, A) := C_q(X)/C_q(A).$$

The chain group $C_q(A)$ must be a normal subgroup of $C_q(X)$ for this quotient to make sense. When using homology with coefficients in an abelian group, then all subgroups are in fact normal and thus we can take quotients.

Relative homology with coefficients in the field $\mathbb{F}$ in the category of simplicial pairs is defined by the homology $\mathrm{H}_q(X, A; \mathbb{F})$ of the chain complex

$$C_q(X)/C_q(A) \xrightarrow{\partial_q} C_{q-1}(X)/C_{q-1}(A),$$

where q is a natural number (including zero) called the index in the chain complex. By definition $\partial_0 := 0$.

Definition 31. The k'th *relative homology* of a simplicial pair (X, A) is given by

$$\mathrm{H}_k(X, A; \mathbb{F}) = \ker \partial_k / \mathrm{im}\, \partial_{k+1}.$$

Whenever $A = \emptyset$, the pair (X, A) is said to be *absolute* and the homology theory will coincide with the absolute homology theories used in applied topology; note the relationship with Definition 30. When using homology with field coefficients, the chain group $C_k(X)$ is defined as the vector space spanned by the k-simplices of X over the field $\mathbb{F}$.

The dual version of homology, called cohomology, is defined by dualizing the chain complex, i.e.

$$C^q(X)/C^q(A) \xleftarrow{\partial_q^* = \delta^{q-1}} C^{q-1}(X)/C^{q-1}(A),$$

where q is the index as before.

Definition 32. The k'th *relative cohomology* is defined as

$$\mathrm{H}^k(X, A; \mathbb{F}) = \ker \delta^k / \mathrm{im}\, \delta^{k-1}.$$

Note that $\delta\delta = 0$ holds since $\partial\partial = 0$. Recall that homology in the category of chain complexes (this includes cochain complexes) requires $\partial\partial = 0$. In the category of abstract simplicial pairs can be achieved by using

Definition 33. The *boundary operator* $\partial_q : C_q \to C_{q-1}$ is defined by

$$\partial_k \sigma = \sum_{i=0}^{k} (-1)^i \sigma | [v_0, \ldots, \hat{v}_i, \ldots, v_k], \tag{5.1}$$

where $\hat{v}_i$ indicates the removal of the i'th vertex and $\sigma|[v_0, \ldots, \hat{v}_i, \ldots, v_k]$ is the restriction of σ to the corresponding face of the simplex.

See [17, pp. 105-106] for a proof that $\partial\partial = 0$.

Homology and cohomology are composite functors from the category of simplicial pairs, factoring through the category of chain complexes, to the category of vector spaces; more generally to the category of abelian groups. It should be remarked that the modelling of a network via abstract simplicial complexes is itself a functor. This functor depends on the object we are modelling.

For example in the case of modeling a communication network, a category is defined as the category with one communication network as object and the identity map as morphism. The functor from this category to the category of simplicial pairs defines how to represent the communication network as an abstract simplicial complex. We could have modelled anything with an interesting topology, e.g., power grid topology etc.

In a 1-dimensional abstract simplicial complex, removing a 0-simplex corresponds to removing a row i in the matrix for ∂_1, and every column which has a non-zero element in row i. For an abstract simplicial complex X, any subset of 0-simplices is a subcomplex $A \subset X$. The following version of Lefschetz duality theorem can be found in [31, p. 297] or [18, p. 178].

Theorem 19 (Lefschetz Duality)**.** *Given an n-dimensional abstract simplicial complex X with a subcomplex A, the following holds*

1. $\mathrm{H}_k(X, A) \cong \mathrm{H}^{n-k}(X - A)$.
2. $\mathrm{H}_{n-k}(X - A) \cong \mathrm{H}^k(X, A)$.

As can be seen in Theorem 19, there is a correspondence between homology and cohomology of simplicial pairs, which relates the k'th homology of a simplicial pair (X, A) with the $(n-k)$'th cohomology of the difference set $X - A$. Likewise the $(n-k)$'th homology of a difference set $X - A$ is related to the k'th homology of a pair (X, A).

5.4. Modelling Networks with Simplicial Pairs

Networks and in particularly communication networks, can be modelled as either being static or dynamic networks. The static networks can most often be modelled via graphs. Obvious examples of static networks would be a transmission grid in power systems or a computer network. These networks are not supposed to change topology in a short time frame under normal operation. Sometimes, however, we need more structure than

a graph can provide and this is where simplicial models are helpful. These models can take an arbitrary but finite dimensionality into account. Unfortunately, simplicial complexes lack one basic flexibility, which is to model more than a single relation between entities in networks, which is not inherited throughout the simplicial structure. Towards solving this issue, we recall that a hypergraph consists of a set of points, called vertices and a collection of sets of points called hyperedges. The category of hypergraphs have a well-known homology theory called Čech homology. It is not as well-behaved as simplicial homology, but luckily there is a nice relation between such a Čech homology and simplicial homology. For a comprehensive introduction to Čech homology, see [12].

Translating a hypergraph to a simplicial complex will be done by considering the so-called nerve of the hypergraph. Note that this in itself can be done because we consider a hypergraph as a special case of a covering.

Definition 34. Let $S = (S_i)_{i \in I}$ be a family of sets where I is an arbitary index set. Then the *nerve* of $S = (S_i)_{i \in I}$ is the simplicial complex $\mathcal{N}(S)$ whose simplices are finite collections of non-empty sets from S with non-empty intersections. Thus the vertices of $\mathcal{N}(S)$ are the non-empty sets from $S = (S_i)_{i \in I}$.

There is one vertex in $\mathcal{N}(S)$ for each set in S. The edges in $\mathcal{N}(S)$ corresponds to pairs (S_i, S_j) in S such that $S_i \cap S_j \neq \emptyset$. Finally the face in $\mathcal{N}(S)$ correspond to the triple (S_i, S_j, S_k) in S such that $S_i \cap S_j \cap S_k \neq \emptyset$. When modelling higher dimensional phenomena where there is an underlying topological space X and the family $S = (S_i)_{i \in I}$ is a family of subsets of X, then it is often easier to consider relations by finite intersections of the subsets rather than directly modelling with simplicial complexes (or even pairs). The corresponding homology theories, however, requires more machinery, which often can be avoided, by applying a relatively simple but powerful lemma, called the *Nerve lemma*, see e.g., [17] for a proof.

Lemma 2 (Nerve lemma)**.** *Let $\mathcal{N}(S)$ be the nerve of some family of subsets $S = (S_i)_{i \in I}$ in a topological space X where all non-empty intersections of subsets from S are contractible. Then $\mathcal{N}(S)$ is homotopy equivalent to X.*

5.5. Topological Ranking

Very often, ranking systems are used to organize and structure systems such that new information can be inferred from already known information. Ranking systems appear in many places and in many variants, see for instance the PageRank algorithm [15]. Unlike many well-known ranking systems, we show that there exists a ranking of objects, which is induced by the shape of the object. This ranking is what we call *topological ranking*.

The main idea is that elements (or sets of elements) which break connectedness, when removed, shall be given a higher ranking. For simplicity, consider a graph X as a one dimensional abstract simplicial complex. The k-simplices of X have varying degrees of importance, in the sense that a given k-simplex α may not have a high rank itself, but combined with other k-simplices, it may achieve a high ranking.

A set of k-simplices $A \subset X^k$ is said to be barrier significant if $\operatorname{rank} \mathrm{H}_0(X) < \operatorname{rank} \mathrm{H}_0(X - A)$. Computation of $\mathrm{H}^1(X, A)$ is straight forward by the associated relative chain complex described in previous sections. Since for a d-dimensional abstract simplicial complex it holds that $\mathrm{H}_0(X - A) \cong \mathrm{H}^d(X, A)$ by Lefschetz duality, the choice of computing $\mathrm{H}_0(X - A)$ or $\mathrm{H}^d(X, A)$ is a matter of convenience. Note that the inclusion map

$$i : (X - A) \hookrightarrow X$$

induces an injective map on homology

$$i_* : \mathrm{H}_0(X - A) \to \mathrm{H}_0(X),$$

if and only if $X - A$ and X have the same number of path components, see e.g. [31] or [17]. With this in mind we make a definition.

The subcomplex A of the abstract simplicial complex X is called a *barrier* if the induced inclusion map $i_* : \mathrm{H}_0(X - A) \to \mathrm{H}_0(X)$ is not injective.

Let X be a 1-dimensional abstract simplicial complex and let A be a 0-dimensional subcomplex. By Lefschetz duality theorem we get that $\mathrm{H}_0(X - A) \cong \mathrm{H}^1(X, A)$. Consider the diagram

$$\begin{array}{ccc} \mathrm{H}_0(X - A) & \xrightarrow{i_*} & \mathrm{H}_0(X) \\ \downarrow \cong & & \downarrow \cong \\ \mathrm{H}^1(X, A) & \xleftarrow{j^*} & \mathrm{H}^1(X). \end{array}$$

The induced inclusion map i_* reveals whether A constitutes a barrier or not. The dual analog j^* is induced by the inclusion map $j : (X, \emptyset) \hookrightarrow (X, A)$. Since we calculate homology with field coefficients, the homology groups are actually vector spaces. Hence the induced inclusion map i_* is a linear map between vector spaces. From this it is clear that A is a barrier if and only if the codimension of the kernel of i_* in $\mathrm{H}_0(X - A)$ is nonzero. In linear algebra, the map i_* may be more tricky than the map j^*, since in the latter case removing rows and columns yield smaller matrices and hence in general faster algorithms. Therefore it may be useful to consider the dual analog j^* in cohomology rather than the induced inclusion map i_* in homology.

In cohomology context, A is a barrier if and only if the codimension of the kernel of j^* in $\mathrm{H}^1(X)$ is nonzero.

Definition 35. A *preorder* of a set $\mathcal{S}$ is a relation in $\mathcal{S}$ which is reflexive and transitive, i.e. a relation $\sim$ in a set $\mathcal{S}$ such that

1. $a \sim a$, for all $a \in \mathcal{S}$ (reflexivity).
2. $a \sim b \ \wedge \ b \sim c \implies a \sim c$, for $a, b, c \in \mathcal{S}$ (transitivity).

A preorder on $\mathcal{S}$ is total if for every $a, b \in \mathcal{S}$ it holds that $a \sim b$ or $b \sim a$. It is now possible to define what is meant by a ranking of a set.

Definition 36 (Ranking)**.** A *ranking* of a set $\mathcal{S}$ is a total preorder $\sim$ on the set $\mathcal{S}$.

Let X be a 1-dimensional abstract simplicial complex. A ranking of the 0-skeleton X^0 of X, called a 0-ranking, is an assignment of numbers to the 0-simplices according to their barrier significance together with the relation $\alpha \sim \beta$ iff. $a_\alpha \leq b_\beta$ where a_α, b_β are the ranks associated to the 0-simplices α and β respectively. Precisely, to each 0-simplex γ of X^0, we associate the *rank* a_γ given as the difference

$$a_\gamma := \dim \mathrm{H}_0(X - \{\gamma\}) - \dim \mathrm{H}_0(X). \tag{5.2}$$

More generally if X is an n-dimensional abstract simplicial complex, then a ranking of the simplices in the s-skeleton X^s of X can be defined, called an s-ranking.

If the cardinality of the finite set of simplices we remove is greater than one, say $k > 1$, then the s-ranking is said to be of order k. In other words, an s-ranking of order k is a ranking of the s-skeleton X^s where the rank associated with each k-tuple $\{\gamma_0, \ldots, \gamma_{k-1}\}$ of simplices in X^s, is given by the difference

$$\dim \mathrm{H}_0(X - \{\gamma_0, \ldots, \gamma_{k-1}\}) - \dim \mathrm{H}_0(X).$$

We get

Proposition 6. *Given an n-dimensional abstract simplicial complex X. Then there exists a topological s-ranking of order k of X, for all k less than the number of simplices in X^s.*

The order of an s-ranking thus defines the number of s-simplices that are removed. The ranking described here will be called *topological ranking*.

5.6. Applications of Topological Ranking

The ranking introduced in this chapter have so many applications, that we can only hope to scratch the surface in this section. In Chapter 6 we consider some more real-life applications. For this section, we consider an example of how to compute topological rankings.

The gossiping problem in information theory is a problem which seeks to decide whether a group of agents, each knowing a unique piece of information, can communicate the accumulated knowledge to everyone in the network, according to a given set of rules. It is more often than not of interest to find the least number of communication instances needed to solve the problem, e.g., the number of packets sent etc. In this section we study in a concrete example the connection between the notion of gossiping in information theory and the topological ranking as introduced in previous sections.

One of the main assumptions underlying the gossiping problem, and other communication problems, is that the communication graph is connected, i.e. all agents are modelled as vertices in the same path-connected component. Topological rankings describes the degree of which this path-connectedness holds, when considering breakage or outage in the communication paths.

Figure 5.1 is interpreted as a model for a gossiping problem of a network with nine agents. The communication protocol is described via a family $\mathfrak{F}$ of four sets S_1, S_2, S_3, S_4 together with the rule that agents can communicate if and only if they belong to the same set. In this case, it is clear that agents in set S_1 can communicate with each other but none of them can communicate with agents in set S_4. For this they need to communicate via agents in S_3 which in turn leads to an elevated connection problem since removing set S_3 will eliminate the possibility to communicate with agents in S_4 altogether.

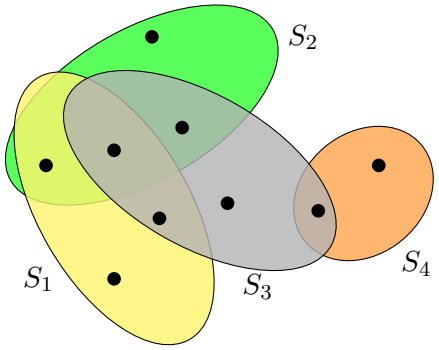

Figure 5.1.: $\mathfrak{F}$ and the nine agents.

For each set in $\mathfrak{F}$ we set make a vertex in a graph. An edge is drawn between two such nodes if the intersection of the corresponding sets is non-empty. A 2-dimensional simplex is drawn if the intersection of a triple of sets in $\mathfrak{F}$ is non-empty. For instance the sets S_1, S_2, S_3 have a common point and thus $S_1 \cap S_2 \cap S_3 \neq \emptyset$.

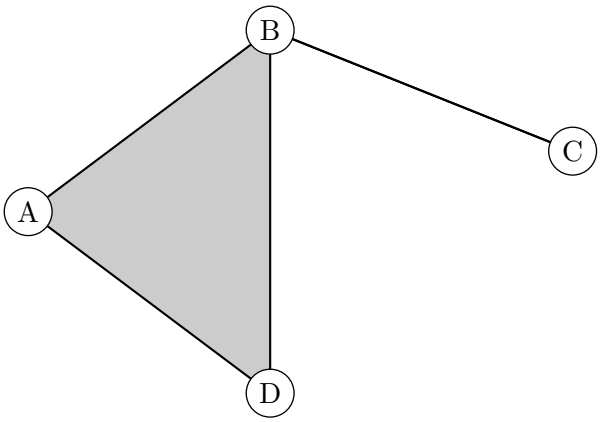

Figure 5.2.: The nerve of $\mathfrak{F}$ and the rank of all elements.

Following the terminology introduced here, whenever two agents belonging to the same set in $\mathfrak{F}$ stop being able to communicate, then the result is that one of them is excluded from the set. When the possibility of communication between two agents from distinct sets in $\mathfrak{F}$ cease to exist it means that the finite intersection of the sets is empty.

For the nerve of the family $\mathfrak{F}$ shown in Figure 5.2, the 0-simplex denoted by B is the only 0-simplex with a non-trivial 0-ranking. The 1-simplex between B and C is the only 1-simplex with a non-trivial 1-ranking.

Let $\mathcal{N}(\mathfrak{F})$ denote the simplicial complex in Figure 5.2. Then the chain complex for $\mathcal{N}(S)$ with coefficient field $\mathbb{Z}_2$ is given by

$$0 \xrightarrow{\partial_3=0} \mathbb{Z}_2 \xrightarrow{\partial_2} (\mathbb{Z}_2)^4 \xrightarrow{\partial_1} (\mathbb{Z}_2)^4 \xrightarrow{\partial_0=0} 0,$$

where ∂_2 and ∂_1 are non-trivial maps. Since homology in dimension zero only depends on the 1-skeleton of $\mathcal{N}(\mathfrak{F})$, the linear map ∂_2 is not relevant with respect to the barrier ranking.

The boundary operator ∂_1 is a linear map between vector spaces over $\mathbb{Z}_2$. After row operations the boundary operator ∂_1 is equivalent to

$$\partial_1 = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

The homology group $\mathrm{H}_0(\mathcal{N}(\mathfrak{F}))$ can then be calculated as

$$\mathrm{H}_0(\mathcal{N}(\mathfrak{F})) = \ker \partial_0 / \operatorname{im} \partial_1 = (\mathbb{Z}_2)^4/(\mathbb{Z}_2)^3 \cong \mathbb{Z}_2,$$

which reflects that the communication graph is connected.

There is only one non-trivial 0-ranking which occurs when we remove the vertex γ in the nerve $\mathcal{N}(\mathfrak{F})$. The rank α_γ of γ is computed by considering the chain complex for the nerve $\mathcal{N}_\gamma$ of the family $\mathfrak{F}$ with the set S_3 corresponding to γ removed. We get the chain complex

$$0 \xrightarrow{\partial_2=0} \mathbb{Z}_2 \xrightarrow{\partial_1} (\mathbb{Z}_2)^3 \xrightarrow{\partial_0=0} 0,$$

in which the boundary operator ∂_1 is given by

$$\partial_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}.$$

From this we find

$$\mathrm{H}_0(\mathcal{N}_\gamma) = \mathbb{Z}_2 \oplus \mathbb{Z}_2,$$

giving the rank of γ

$$\alpha_\gamma = \operatorname{rank} \mathrm{H}_0(\mathcal{N}_\gamma) - \operatorname{rank} \mathrm{H}_0(\mathcal{N}(\mathfrak{F})) = 1.$$

5.7. Future Research: Stanley-Reisner Rings and Koszul Complexes

One of the consequences of the inherent topological ranking underlying all simplicial structures in network theory, is that once we settle for a topology, then our networks have fixed induced weaknesses. While these weaknesses can be minimized, it is unknown (at least to the author) if there exists a way to determine these weaknesses in a reasonable amount of time for it to be practical. It is very unlikely that faster algorithms can be found without introducing minimal free resolutions and so-called Koszul complexes.

There exists a functor from the category of simplicial complexes to the category of commutative rings, which we shall call the Stanley-Reisner construction. The main idea is to transform a simplicial complex into a polynomial ideal (over a field) in such a way that certain topological information is retained in the polynomial ring. The construction is reversible in the sense that we can construct a simplicial complex from certain polynomial ideals. The construction gives rise to a ring which is called the Stanley-Reisner ring or the face ring.

Definition 37. Given a finite dimensional simplicial complex X with zero simplices $V = \{x_0, \ldots, x_t\}$. The Stanley-Reisner ring (or face ring) is defined to be the quotient ring $\mathbb{F}[X] = \mathbb{F}[x_0, \ldots, x_t]/I_X$, where

$$I_X = \langle x_{i_1} x_{i_2} \ldots x_{i_r} \mid i_1 < i_2 < \cdots < i_r \,,\ \{x_0, \ldots, x_{i_r}\} \notin X \rangle.$$

Consider the simplicial complex X shown in Figure 5.3.

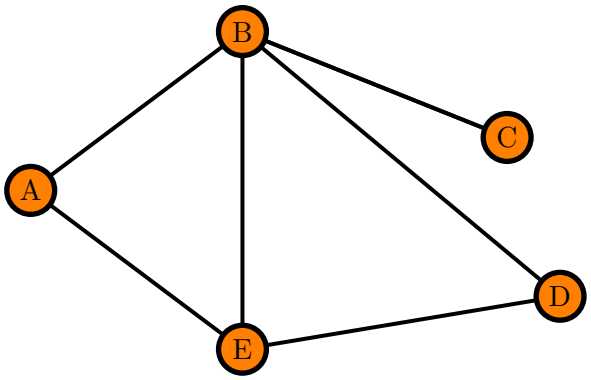

Figure 5.3.: Simplicial Complex

The Stanley-Reisner ring is the quotient $R = \mathbb{F}[A, B, C, D, E]/I_X$, where

$$\begin{aligned} I_X &= \langle AC, AD, CD \rangle \\ &= \langle A, C \rangle \cap \langle A, D \rangle \cap \langle C, D \rangle. \end{aligned}$$

The main advantage that the author hopes for is that instead of finding the collections of simplices which constitute a barrier, it would be very useful to find a construction, which gives some indication as to where to look for barriers.

5.7.1. Alternative Boundary Formula

One of the main building blocks for designing a homology theory via the category of chain complexes, is the definition of boundary operators used in each object of the category of chain complexes. We used a well-known boundary operator for the simplicial homology, but if we consider more complicated structures such as free resolutions, we need to redefine this operator. There is not a well-known unique way of doing this. We show one way of defining a boundary operator for polynomial rings.

We define a derivation $D : \mathbb{F}_2[x_1, \ldots, x_n] \to \mathbb{F}_2[x_1, \ldots, x_n]$ that is suitable as a boundary operator in homology theory. Since we are in the case of polynomials with binary coefficients, we define D to be the the sum of partial differentials, known from calculus, i.e.,

$$Df = \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}, \quad f \in \mathbb{F}_2[x_1, \ldots, x_n].$$

Lemma 3. *Let $D : \mathbb{F}_2[x_1, \ldots, x_n] \to \mathbb{F}_2[x_1, \ldots, x_n]$ be defined as above. The boundary of a boundary is zero in $\mathbb{F}_2[x_1, \ldots, x_n]$, i.e. $D^2 = 0$.*

Proof. It is clear that $D^2(x_i^t) = 0$ for $x_i^t \in \mathbb{F}_2[x_1, \ldots, x_n]$, $t \in \mathbb{N}$. Furthermore, D is a linear map. For consider

$$D(f+g) = \sum_{i=1}^{n} \frac{\partial}{\partial x_i}(f+g) = \sum_{i=1}^{n} \left(\frac{\partial}{\partial x_i} f + \frac{\partial}{\partial x_i} g \right) = \sum_{i=1}^{n} \frac{\partial}{\partial x_i} f + \sum_{i=1}^{n} \frac{\partial}{\partial x_i} g.$$

Therefore it will suffice to consider monomials in $\mathbb{F}_2[x_1, \ldots, x_n]$. Consider

$$\begin{aligned} D^2(x_1^{t_1} \cdots x_n^{t_n}) &= D \sum_{i=1}^{n} \frac{\partial}{\partial x_i}(x_1^{t_1} \cdots x_n^{t_n}) \\ &= \sum_{j=1}^{n} \sum_{i=1}^{n} \frac{\partial^2}{\partial x_j \partial x_i}(x_1^{t_1} \cdots x_n^{t_n}). \end{aligned}$$

Since we can consider the polynomial $f \in \mathbb{F}_2[x_1, ..., x_n]$ as a differentiable function of class C^2 in $\mathbb{R}^n$, it is well known that

$$\frac{\partial^2 f}{\partial x_j \partial x_i} = \frac{\partial^2 f}{\partial x_i \partial x_j}.$$

Thus all monomials in $D^2(x_1^{t_1} \cdots x_n^{t_n})$ are either zero or present in the sum, twice. Therefore $D^2 = 0$.

□

In order to make this boundary operator useful for homology, we would need to choose the chain complexes appropriately.

Chapter 6

Optimization of Network Design

From the previous chapters it can be argued that homology theory and the theory of Gröbner bases can be applied in a wide area of disciplines both within mathematics and in the engineering sciences. While graph vertex covering can be approximated, say by the *greedy algorithm*, it is sometimes required that we find the exact solutions and maybe not just *a* solution but *all* solutions. For instance, the well-known problem of assigning frequencies to GSM networks, the Global System for Mobile Communications, involves using four frequency bands and assign these into each area, i.e. tower, such that no two adjacent areas have the same frequency; thereby avoiding interference. In this example it will not suffice to approximate a solution since it would be unacceptable to have interference in any region. The interesting questions from an engineering point of view, is not whether there exists an efficient solution, which is unlikely, but rather how many regions can we solve this problem for. The *performance* is the key indicator for applying a methodology to solve a problem. While from the purely theoretical perspective, the economics may not always be important. In real-life applications, such as building research facilities or operating power grids, the economics is a key factor. Again, the exactness of a solution may help save cost by finding minimum installations needed to achieve a certain goal.

Recent studies have considered the connection between the minimal phasor measurement unit (PMU) placement and commutative algebra by stating bounds on the minimal PMU covers via calculating the Krull dimension of quotient rings, see [7, 26]. These methods are very useful, and would be a great place to start mixing methods developed in this thesis with classical methods from both electrical engineering and graph theory.

This chapter will focus on applying the topological ranking algorithm developed in Chapter 5 and the parallel algorithm for computing the intersection of square-free monomial ideals, developed in Chapter 4. We use these methods to illustrate how interesting questions can be posed for bus systems in power grids. All networks considered here can be substituted with other networks in order to find different applications.

6.1. Edge and Cover Ideals of Hypergraphs

We shall introduce the two definitions which makes our study of parallel intersection of monomial ideals useful in the context of PMU placement. For hypergraphs (and thus graphs) we can introduce square-free monomial ideals which contain much information known from graph theory.

Definition 38. Given a hypergraph H, the *path (edge) ideal* is defined as

$$I_H = \left\langle x_e = \prod_{x \in e} \mid e \in E \right\rangle.$$

There is nothing difficult in constructing edge ideals, since the process of constructing them is deterministic. This is in contrast to the cover ideals of hypergraphs, which we shall define shortly.

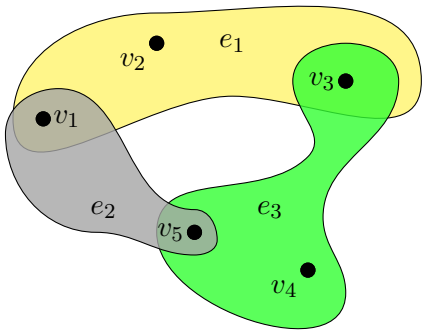

Figure 6.1.: Hypergraph

Example 21. Consider the hypergraph in Figure 6.1 and denote it by H. Let the variables $x_1, \ldots, x_5$ correspond to the vertices. Fix the monomial order to be the lexicographic ordering $x_1 > \cdots > x_5$. Then define the edge ideal $I_H \subset \mathbb{F}_2[x_1, \ldots, x_5]$ over the polynomial ring with coefficients in $\mathbb{F}_2$ by

$$I_H = \langle x_1x_2x_3, x_1x_5, x_3x_4x_5 \rangle.$$

Our main interest here is the notion of a cover ideal, which contains information about the vertex covers for hypergraphs (and thus graphs). We see that cover ideals are computed using intersections, which we know an algorithm for computing in parallel.

Definition 39. The *cover ideal* $J \subset \mathbb{F}[x_1, \ldots, x_n]$ of a hypergraph H with vertex set $V = \{v_1, \ldots, v_n\}$ and edge set $E = \{e_1, \ldots, e_t\}$, is given by

$$J_H = \bigcap_{e \in E} \langle x \mid x \in e \rangle.$$

Example 22. Again, we model the hypergraph in Figure 6.1 as in Example 21. The cover ideal is given by

$$\begin{aligned} J_H &= \langle x_1, x_2, x_3\rangle \cap \langle x_1, x_5\rangle \cap \langle x_3, x_4, x_5\rangle \\ &= \langle x_1, x_3x_5, x_2x_5\rangle \cap \langle x_3, x_4, x_5\rangle \\ &= \langle x_3x_5, x_2x_5, x_1x_5, x_1x_4, x_1x_3\rangle. \end{aligned}$$

Notice that the generators of J_H correspond exactly to the minimal vertex covers of the hypergraph H. Each generator is a vertex cover.

Remark. The cover ideal will sometimes contain generators which are not "minimal", in the sense that the corresponding vertex cover is not a minimal vertex cover. However, the cover ideal will always contain all minimal vertex covers.

The size of the minimal vertex cover of a graph can be stated in terms of the Krull dimension.

Theorem 20 ([26])**.** *Let $G = (V, E)$ be a graph, with vertex set $V = \{v_1, \ldots, v_n\}$. Then the Krull dimension is given by*

$$\dim(\mathbb{F}_q[x_1, \ldots, x_n]/I_G) = n - d,$$

where I_G is the edge ideal of G, and d is the size of the smallest vertex cover for G.

It seems that this theorem can be generalized to include hypergraphs.

Conjecture 4. *Let H be a hypergraph, with vertex set $V = \{v_1, \ldots, v_n\}$. Then the Krull dimension is given by*

$$\dim(\mathbb{F}_q[x_1, \ldots, x_n]/I_H) = n - d,$$

where I_H is the edge ideal of H, and d is the size of the smallest vertex cover for H.

In support of this conjecture we give

Example 23. Define the hypergraph H and corresponding edge and cover ideal, I_H and J_H resp., to be the same as in Examples 21 and 22. Consider $R = \mathbb{F}_2[x_1, \ldots, x_5]/I_H$.

By considering J_H it is easy to see that the size of the minimal vertex cover is 2 and if Conjecture 4 is true, then the Krull dimension of R is

$$\dim(\mathbb{F}_2[x_1, \ldots, x_5]/I_H) = 5 - 2 = 3.$$

Verifying on a computer, e.g., using Macaulay2 yields the same result.

Remark. It has not been tested, how efficient the algorithms are, for computing quotient rings and their Krull dimension using Macaulay2 (or any other software). Therefore it is currently not known to the author, if this can be used for determining a bound for the size of the minimal vertex cover.

6.2. Phasor Measurement Unit Placement

A phasor measurement unit (PMU) is a device that can be placed at a *bus* to measure voltage at the bus and the current of all transmission lines connected to that bus. Placing these PMUs is known at the PMU placement problem. This problem has been studied using many methods such as integer programming, graph theory, heuristic methods and Gröbner bases. We apply Gröbner bases to find minimal vertex covers and use topological ranking to choose minimal vertex covers such that grid normalization, after outages, can be done in a way that makes the normalized grid observable.

A bus is said to be *observable* if the voltage at the bus is known. A PMU cover is a placement of PMUs such that every bus in the electrical bus system is observed. This notion is almost indistinguishable from that of a vertex cover in graph theory. A PMU cover is said to be *minimal* if it is impossible to observe the power system with fewer PMUs.

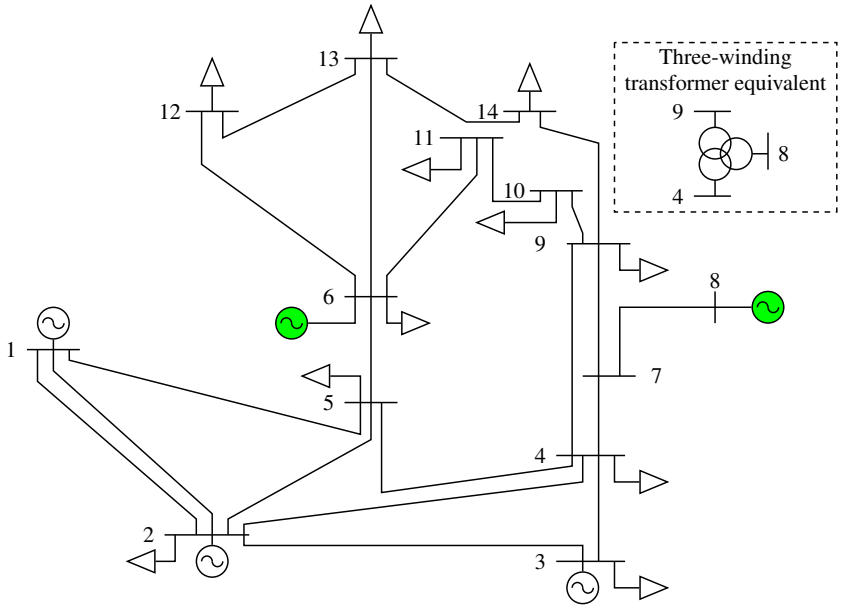

Figure 6.2.: IEEE 14 Bus System

With reference to Figure 6.2, the buses $1, 2, 3, 6, 8$ are connected to generators. There are 11 loads, which are located at buses $2, 3, 4, 5, 6, 9, 10, 11, 12, 13, 14$. The bus system can be described, via a forgetful functor, as an object in the category of graphs (note that morphisms must be treated separately for vertices and edges).

In order to observe the IEEE 14 bus system, one could among different choices, place PMUs at the buses,

Bus 1, Bus 2, Bus 4, Bus 6, Bus 7, Bus 9, Bus 10, Bus 13.

The added PMUs in Figure 6.3 represent a minimal vertex cover in the graph model of the bus system. The number of required PMUs is obviously lower than placing PMUs

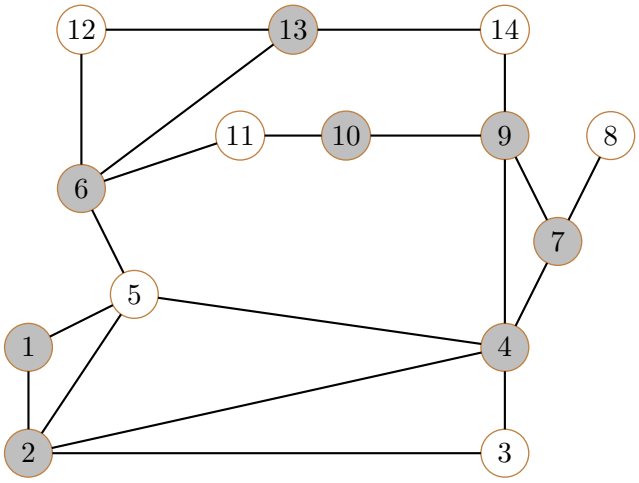

Figure 6.3.: IEEE 14 bus system topology

at every bus. Some authors have found other methods which could complement this approach to reduce the number of required PMUs even further.

By utilizing square-free monomial ideals to solve the PMU placement problem, we are in the situation where, sometimes, a partition of graphs can be used to induce a covering of each partition such that the addition of new connections, e.g., backup connections, the resulting graph has a minimal vertex cover without adding new PMUs. This idea can be applied when designing contingency plans for power grid operation failures or planned maintenance shutdown procedures.

One interesting partition that can be applied on a bus system, is via the topological ranking defined in Chapter 5. For instance if we consider the maximum topological 0-ranking of order 2 of the IEEE 14 bus system, we get the following critical sets,

$$\{4,6\},\ \{6,9\},\ \{4,5\},\ \{5,9\}.$$

We quickly observe that buses $4,5,6,9$ are, from a purely topological perspective, among the most critical. It is noted that most of these sets overlap with the cover chosen in Figure 6.3. If needed we can choose a different vertex cover (and thus PMU placement) in order to reduce this overlap. The advantage of not overlapping the PMU placement and the "critical" nodes, we can get a network design which, in case of outages, is able to retain more of its observable features when backup lines are taken into account. Figure 6.4 show the consequence of choosing the vertex cover in Figure 6.3. The problems that may arise, are seen in the choice of backup connections. For instance we may want to place a backup line between buses 5 and 11. With the current vertex cover, namely at buses $1,2,4,6,7,9,10,13$, we see that the backup line between 5 and 11 will yield a network were the observability is questionable, see Figure 6.4.

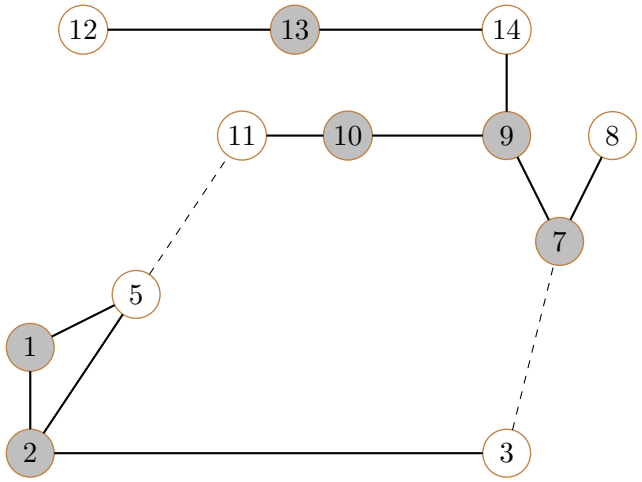

Figure 6.4.: IEEE 14 bus system topology with outage at 4 and 6

The vertex cover shown in Figure 6.5 illustrates a minimal cover, which given backup lines between 5 and 11 and between 3 and 7 will be fully observable, see Figure 6.6.

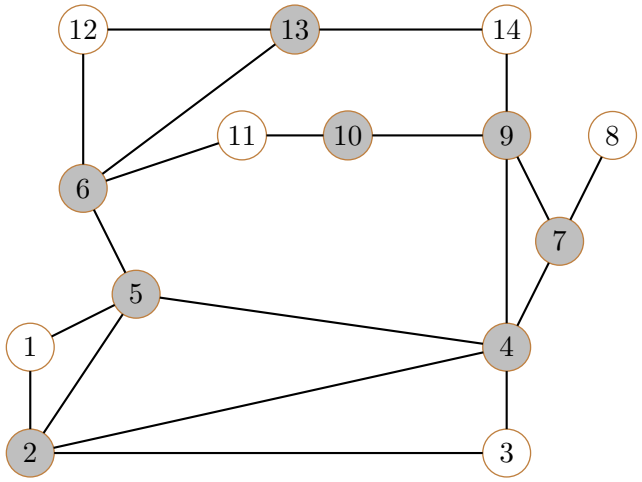

Figure 6.5.: IEEE 14 bus system topology with outage at 4 and 6

Choosing the vertex cover in Figure 6.5 instead of that in Figure 6.3, we see that the 0-ranking of order 2, does not interfere in the same way. In particular, it can be observed that the number of operational PMUs can be maximized if we choose a proper minimal vertex cover. If for instance we have failures at buses 4 and 5. Then the bus system is separated and we are left with two disjoint graphs.

While we can see that no matter how we remove a set of vertices (or edges) in a graph, a vertex cover of the original graph will still be covering the graph after removal, possibly by yielding a covering for each path-connected component.

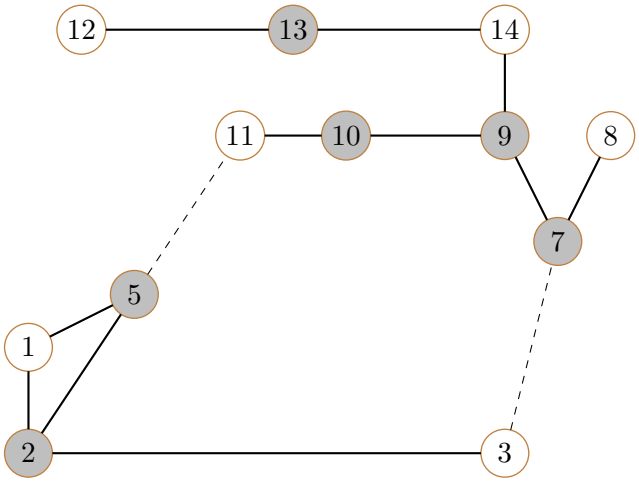

Figure 6.6.: IEEE 14 bus system topology with outage at buses 4 and 6

6.2.1. Mandatory Placement

It is often the case, when working with the development of network systems that some structure is already known or required. In the case of electrical bus systems and more specifically for PMU placement, we might already know that a given set of buses should hold PMUs or they may already have PMUs installed.

In such a situtation, we are not interested in all the coverings which do not adhere to this reality. Working with monomial ideals as models for vertex covering, we are able to model this very efficiently. Given a cover ideal J, e.g., for the IEEE 14 bus system, we can choose only the coverings correpsonding to those which place some of the PMUs at very specific locations. For the IEEE 14 bus system, we can for instance require that bus 6 gets a PMU by considering $J \cap \langle x_6 \rangle \in \mathbb{F}[x_1, \ldots, x_{14}]$, where J is the cover ideal of the bus system and the index of x_i corresponds to the bus number in the 14 bus system. The result is a set of vertex coverings which contain x_6, equivalently places a PMU at bus 6.

If we have a set of buses that either have a PMU installed already or there are plans for installing a PMU, it is possible to restrict or intersect even further. If the set $S = \{x_{i_1}, \ldots, x_{i_t}\}$ is a set of mandatory placements, then we need to consider successive intersections of principal ideals $\langle x_{i_s} \rangle$ with the original cover ideal. We get the following cover ideal with mandatory placements

$$J \cap \langle x_{i_1} \rangle \cap \cdots \cap \langle x_{i_t} \rangle.$$

Since intersections of ideals is associative, it is possible to define these very strict rules in the beginning of the calculation of the cover ideal. It is unknown if this will yield improvements of the computations; the author suspects that improvements can be made by computing intersections with principal ideals as part of the initial calculations. It is however speculation at this point.

Figure 6.7 show the IEEE 14 bus system where the vertex covering has mandatory placement at vertices $1, 3, 12$ and 14. The minimal number of PMUs is no longer 8 but 9 PMUs are required to satisfy this constraint.

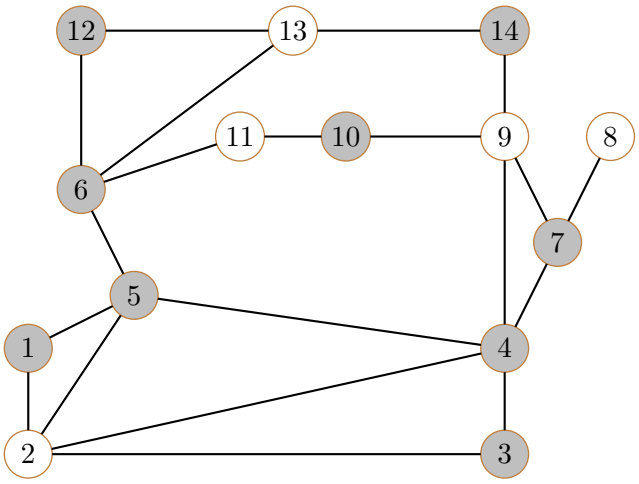

Figure 6.7.: IEEE 14 bus system topology with mandatory placement

6.2.2. Mutual Exclusion Zones

Imagine that we define an area of a network, e.g., a set of nodes, which only a single PMU is required among each set of nodes. This can be modelled using a hypergraph and thus hyperedges encode information about exclusivity rules. For instance in the IEEE 14 bus system, we might make an additional requirement that only one of the buses $3, 11, 12, 14$ needs to have a PMU installed. This will make the analysis more flexible in the sense that more sophisticated rules can be made. Such a set of buses will be called a *matual exclusion zone*.

These rules will be presented by constructing a hyperedge containing the set of buses, i.e. a mutual exclusion zone is a hyperedge. In the IEEE 14 bus system, assume that $\{3, 11, 12, 14\}$ is a mutual exclusion zone. The hyperedge representing this mutual exclusion zone is then mapped to a monomial ideal in the following way: For each bus in the mutual exclusion zone, equivalently for each vertex in the hyperedge corresponding to the mutual exclusion zone, define a principal monomial ideal containing the variable representing the particular bus. We then add the principal monomial ideals and the resulting monomial ideal is added to the sequence of ideals used to calculate the cover ideal. If $J \subset \mathbb{F}[x_1, \ldots, x_n]$ is the cover ideal of the IEEE 14 bus system topology. Then taking the mutual exclusion zone $\{3, 11, 12, 14\}$ into account yields

$$J \cap \langle x_3, x_{11}, x_{12}, x_{14} \rangle.$$

One of the minimal vertex covers taking the mentioned mutual exclusion zone into account is $x_2x_4x_5x_6x_8x_9x_{11}x_{13}$. Note that bus 11 is the only bus from the mutual exclusion zone.

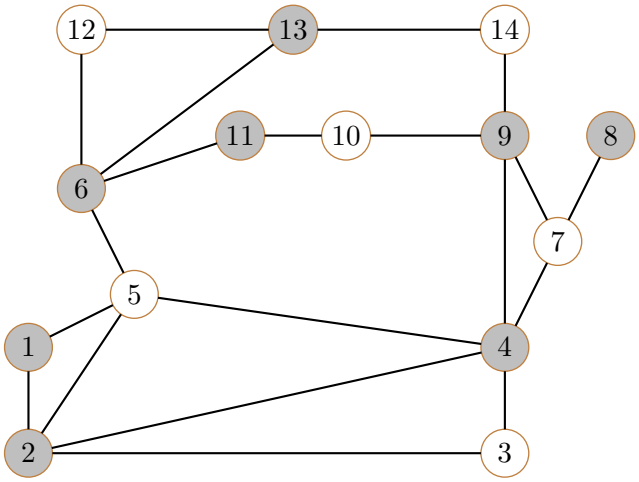

Figure 6.8.: IEEE 14 bus system topology with mutual exclusion zone

Figure 6.8 shows a minimal covering for the IEEE 14 bus system with a mutual exclusion zone defined to be $\{3, 11, 12, 14\}$. Mutual exclusion zones guarantees that at least on bus in the zone will get a PMU. Furthermore the calculation of the cover ideal will seek to minimize the number of PMUs to be placed in the mutual exclusion zone.

6.2.3. IEEE 30 Bus System

For larger bus systems, it will inevitably become less obvious where to place PMUs in order to achieve a minimal coverage. The methods developed in this thesis, will be used to calculate all minimal vertex coverings for the IEEE 30 bus system and list some of them in a table. There are 368 distinct minimal vertex coverings for the IEEE 30 bus system and it is required to use at least 16 PMUs to cover the system topologically.

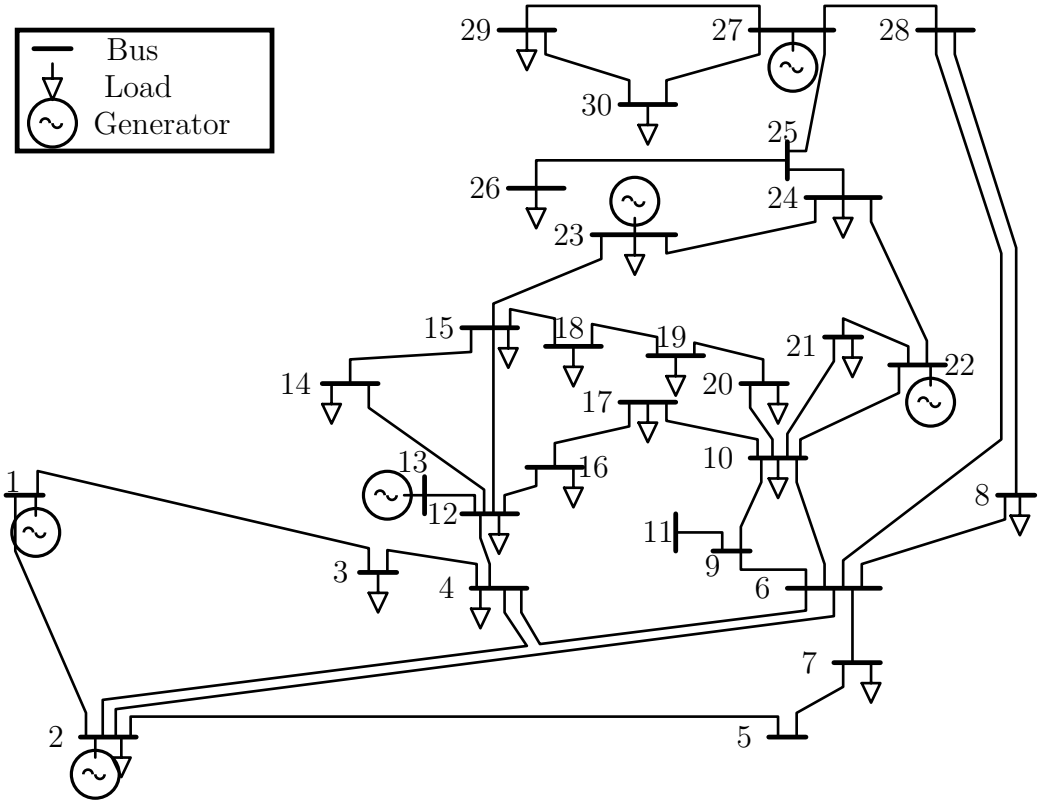

Figure 6.9.: IEEE 30 Bus System

We highlight some possible minimal PMU placements for the IEEE 30 bus system in table 6.1. Though there are locations which are contained in all of the shown placements, this is merely a result of showing only a subset of the possible placements.

If we for example choose to have mandatory PMU placement at buses $2, 13, 22$ and 27, then there are 96 minimal coverings using at least 17 PMUs. The increase in the number of required PMUs can be seen on Figure 6.10 at the connection between buses 12 and 13, where PMUs are placed redundantly which is enforced by the mandatory placement at bus 13. The thickness of the buses indicate a PMU is placed at the bus, e.g., bus 13 is modelled using a thick line to indicate placement of a PMU.

PMU bus locations
3,4,5,6,8,10,11,12,15,17,19,22,24,25,27,29
1,4,5,6,8,10,11,12,15,17,19,22,24,25,27,29
2,3,5,6,8,10,11,12,15,17,19,22,24,25,27,29
2,3,6,7,8,9,10,12,15,17,19,22,24,25,27,29
3,4,5,6,8,9,10,12,15,17,19,22,24,25,27,29
1,4,5,6,8,9,10,12,15,17,19,22,24,25,27,29
2,3,5,6,8,9,10,12,15,17,19,22,24,25,27,29
2,3,6,7,8,10,11,12,15,16,19,22,24,25,27,29
3,4,5,6,8,10,11,12,15,16,19,22,24,25,27,29
1,4,5,6,8,10,11,12,15,16,19,22,24,25,27,29
2,3,5,6,8,10,11,12,15,16,19,22,24,25,27,29

Table 6.1.: Possible minimal coverings of IEEE 30 bus system

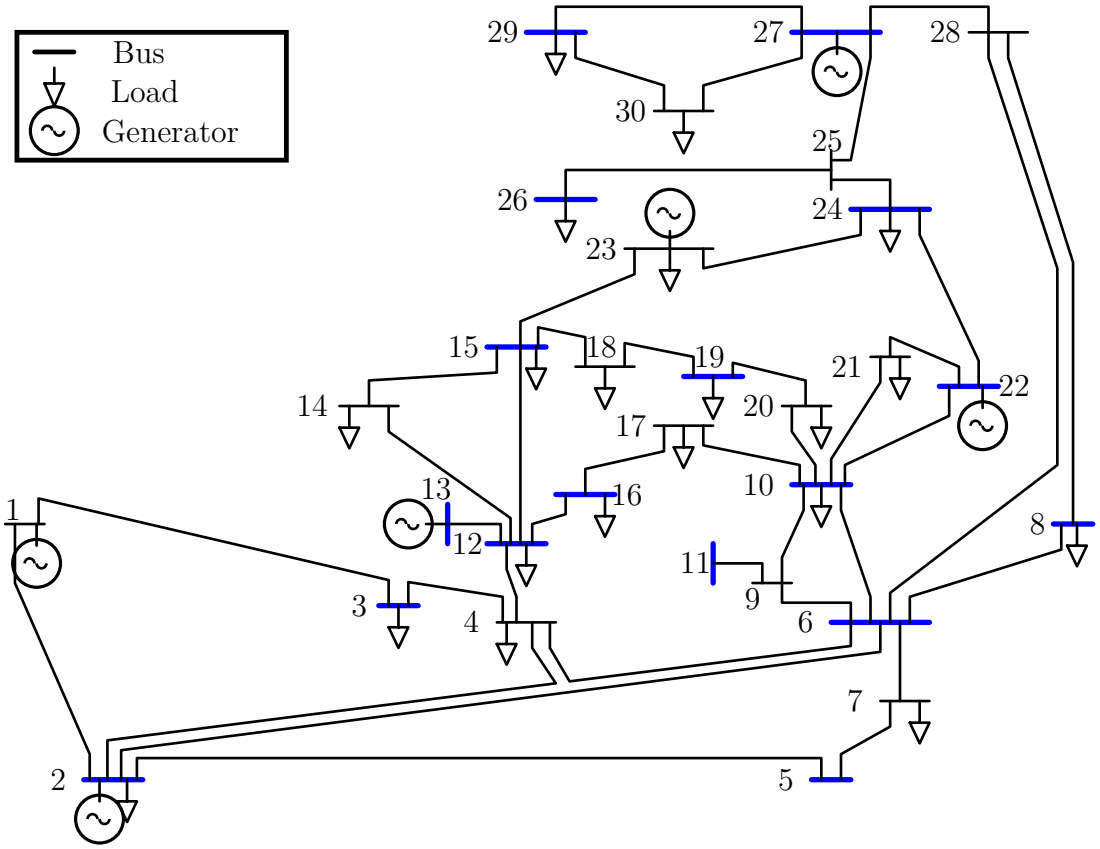

Figure 6.10.: IEEE 30 Bus System with mandatory placement of PMUs

PMU bus locations with mandatory placement
2,3,6,7,8,9,10,12,13,15,17,19,22,24,26,27,29
2,3,5,6,8,9,10,12,13,15,17,19,22,24,26,27,29
2,3,6,7,8,10,11,12,13,15,16,19,22,24,26,27,29
2,3,5,6,8,10,11,12,13,15,16,19,22,24,26,27,29
2,3,6,7,8,9,10,12,13,15,16,19,22,24,26,27,29
2,3,5,6,8,9,10,12,13,15,16,19,22,24,26,27,29
2,3,6,7,8,10,11,12,13,15,17,19,22,24,25,27,29
2,3,5,6,8,10,11,12,13,15,17,19,22,24,25,27,29
2,3,6,7,8,9,10,12,13,15,17,19,22,24,25,27,29
2,3,5,6,8,9,10,12,13,15,17,19,22,24,25,27,29
2,3,6,7,8,10,11,12,13,15,16,19,22,24,25,27,29

Table 6.2.: Minimal coverings of IEEE 30 bus system with mandatory placement

Table 6.2 shows a small number of placements associated with the mandatory placement at buses 12 and 13.

6.3. Concluding Remarks

Combining topological rankings and cover ideals, can be useful when designing grid topology or other network related design. Great care must be taken whenever decisions are made, which influence the topology or the derived characteristics. Some decisions can have far reaching consequences for the operation of the system. By considering mandatory placement, we are restricting the set of generators of the initial cover ideal, e.g., the IEEE 14 bus system have 16 minimal vertex coverings, placing only 8 PMUs, and mandatory placement of PMUs on buses $4, 5, 6$ and 9, yields only 4 minimal coverings using 8 PMUs. Likewise considering mutual exclusion zones we can get more choices for minimal vertex covers. For instance if we consider the mutual exclusion zone $\{3, 11, 12, 14\}$, then there are 18 choices for minimal vertex covering using 8 PMUs. This is in contrast to the original 16 distinct minimal coverings.

If we combine the mandatory placement, mentioned above, and the mutual exclusion zone, also mentioned above, we get that there are 6 minimal vertex coverings using 8 PMUs. Combining methods developed in this dissertation together with methods already published in this area may yield interesting new results and insights.

In the same spirit as the development of the topological ranking, it is suspected that we can define another kind of ranking based on minimal vertex coverings via mandatory placement. If a mandatory labelled vertex increases the minimal number of PMUs required to cover the graph, then we can define the rank of that vertex to be the difference in the minimal number of PMUs, before and after.

Chapter 7

Discussion and Conclusion

In this dissertation, contributions have been made to algebraic geometry and mathematical modelling of problems arising in network theory. The underlying theme has been algebraic varieties and though not all chapters seem to be directly related to these objects, it is possible to conceive everything in this dissertation as problems involving algebraic varieties in the category of commutative rings or its opposite counterpart, the category of schemes. The (co)homology theory discussed in Chapter 5 is calculated using the category of vector spaces and linear maps, and this can be seen as a full sub-category of the category of commutative rings. Moreover the homology classes (and cohomology classes) are thus algebraic varieties. When we compute homology with coefficients in a finite field, the algebraic varieties are considered over these finite fields. For instance when calculating homology with coefficients in the finite field $\mathbb{F}_q$, we functorially transfer the problem to a problem of solving a set of linear equations over $\mathbb{F}_q$ in the category of vector spaces. Every equation in such a system can be viewed as a generator of a polynomial ideal in a suitably chosen polynomial ring $\mathbb{F}_q[x_1, \ldots, x_n]$. We intentionally refrained from taking this latter viewpoint in favor of using the traditional approach using classical linear algebra. The reasons for this was merely that it was unclear if any benefit would come from taking a more abstract approach.

The well-known Euler's criterion was generalized and extended. It was given a formulation which makes it possible to use it, in the construction and analysis of finite field extensions. The original Euler's criterion is already heavily used in many areas of mathematics and engineering and it is likely that the formulation given here will be useful in some of these areas of application.

The circle equation over finite fields was studied extensively and has served as a model example for analyzing algebraic varieties over finite fields. It became apparent during the study of this canonical equation, that studying algebraic varieties over finite fields have far reaching connections with number theory. This led to the definition of what Vagn and I call siameese twin primes and a conjecture that all twin primes can be divided into two equally large sets, partitioned into siameese twin primes and non-siameese twin primes. The conjecture is supported for an incredibly large number of twin primes and the conjecture depends on and extends the famous twin prime conjecture.

An algorithm for computing the intersection of monomial ideals have been developed

and implemented. While it comes as a surprise to no one that taking a sequence of any kind of intersection can be done in parallel, it was peculiar to observe that the order of which these intersections were done greatly impacted the performance. Fortunately we were able to resolve this problem by defining a means of sorting the sequence prior to computing the intersections. The result was a scalable and performance efficient parallel algorithm for computing vertex covers of hypergraphs.

We have utilized a well-known homology theory to show that the choice of (simplicial) topology induces a ranking of all entities (simplices) in the simplicial structure. We term this total ordering the topological ranking of the simplicial complex in question. The ranking is induced by removing a set of simplices.

The last part of the dissertation discussed how to combine the topological ranking with the calculation of vertex covers of a 1-dimensional simplicial complex (possibly the nerve of a hypergraph) to highlight certain problems in power grid topology design. One of the strengths of using square-free monomial ideals to calculate the vertex covers was that all minimal vertex covers were calculated at the same time; one vertex cover for each generator in the cover ideal with lowest total degree. Similar considerations could have been made for the calculation of the topological ranking but time did not permit the development of such a framework for calculating the ranking. Nevertheless combining the ranking with the minimal vertex covers reveals alternative ways to construct contingency plans for operating power grids. The proposed methodology is very flexible and can accomodate many real-life problems such as PMUs already installed in possibly sub-optimal locations in the system etc.

Among the various problems that the author will be interested in considering in the future are studying the siameese twin primes which may yield more insight into the twin prime conjecture. Further the computation of (reduced) Gröbner bases of specific polynomial ideals may provide new results on the complexity of Gröbner basis computation, while varying the base field. We can only hope that this will give us extended insights into the millennium problem, $P = NP$; a problem which the majority of the scientific community believes to be false. The author remains agnostic on the truthfulness of this statement.

Probably the most accessible future research will be the computation of the topological ranking, possibly by introducing a specialized homology theory. The author conjectures that there exists a homology theory which combines both the topological ranking and the vertex cover problem into a single and highly optimized framework. Such a framework requires clearly defined ontologies to be described and it is very much possible that such a homology theory will be a variant of the highly abstract, but extremely useful l-adic cohomology. This cohomology theory was pioneered by Alexander Grothendieck with the help of Jean-Pierre Serre and Michael Artin. It was introduced in an effort to prove the famous Weil conjectures, the last of which was proved by Pierre Deligne using l-adic cohomology.

The author hopes to further develop a unifying methodology for calculating topological rankings and computing cover ideals.

Appendix A

Haskell Code for Working with the Circle Equation

Several functions and programs were developed during the investigation of the circle equation over finite fields. Some of the most useful programs will be printed in this appendix.

To start with, we work over positive characteristic and thus we need to be able to find many prime numbers. The fastest methods use a so-called *sieve* and in Haskell this can be implemented like this.

```
-- prime number generator
primes :: [Integer]
primes = 2: 3: sieve (tail primes) [5,7..]
  where
      sieve (p:ps) xs = h ++ sieve ps [x | x <- t, x `rem` p /= 0]
        where (h,~(_:t)) = span (< p*p) xs
```

Another elementary functionality which is often needed, is to determine if a given positive integer is a prime number.

```
divides :: Integral a => a -> a -> Bool
divides d n = rem n d == 0

ldf :: Integral a => a -> a -> a
ldf k n | divides k n = k
        | k*k > n = n
        | otherwise = ldf(k+1) n

ld :: Integral a => a -> a
ld n = ldf 2 n

prime0 :: Integral a => a -> Bool
prime0 n | n < 1     = error "not a positive integer"
         | n == 1    = False
         | otherwise = ld n == n
```

In our numerical investigation of the number of solutions to the circle equation over finite fields, it was needed to remove duplicates in a list, for evident reasons. Note that this function requires the package *Data.List*.

```
-- remove dublicates in list
rmdups :: (Ord a) => [a] -> [a]
rmdups = map head . group . sort
```

One of the very nice features of Haskell is the set comprehension. We used for, among other things, finding primes in a given integer interval $[n; m]$ or in the *sieve* method for finding primes illustrated above.

```
-- find primes in an interval
findPrimes n m = [ x | x<-[n..m], prime0 x == True]
```

Finding twin primes can be done using set comprehension as well. The following function returns the n first twin primes utilizing the sieve method for finding primes.

```
-- grab the first n twin primes
twinPrimes :: Int -> [(Integer, Integer)]
twinPrimes n = [ (x,y) | x <- (take n primes),
  y<-(take n (tail primes)), x+2==y, x<y]
```

One of the very useful tools that we have used was the quadratic character homomorphism. In the following we assume that the input is an appropriately defined finite field type in the category of Haskell types.

```
-- quadratic character map
qf :: Num t => [t] -> Integer -> [t]
qf gf r = [ x^r | x <- gf]

eta :: (Num a1, Num a, Eq a1) => [a1] -> a1 -> Integer -> a
eta gf c r = if (c /= 0 && any (c==) (qf gf r)) then 1
  else if (c==0) then 0 else -1
```

Notice that the function qf and eta can easily be changed such that it computes the extended Euler's criterion.

Appendix B

Manuscript for Article: The Circle Equation over Finite Fields

The article titled *The Circle Equation over Finite Fields* co-authored with Vagn Lundsgaard Hansen is a manuscript in preparation.

THE CIRCLE EQUATION OVER FINITE FIELDS

ANDREAS AABRANDT AND VAGN LUNDSGAARD HANSEN

Abstract. Interesting patterns in the geometry of a plane algebraic curve C can be observed when the defining polynomial equation is solved over the family of finite fields. In this paper, we examine the case of C the classical unit circle defined by the circle equation $x^2 + y^2 = 1$. As a main result, we establish a concise formula for the number of solutions to the circle equation over an arbitrary finite field. We also provide criteria for the existence of diagonal solutions to the circle equation. Finally, we give a precise description of how the number of solutions to the circle equation over a prime field grows as a function of the prime.

Subject class: 11G20, 11D45, 11A07, 14G15
Keywords: Diophantine geometry, prime numbers, siamese twin primes

1. Introduction

From ancient time, shapes and numbers have been fundamental objects for organizing any kind of civilization, and the birth of mathematics is intimately related to exploring these objects. In the Greek culture, studies of shapes and numbers went hand in hand and culminated in work of Diophantus in the third century. Diophantus has lent his name to *diophantine geometry*, which is the study of geometrical properties of the set of solutions to polynomial equations over integers, rational numbers and more general number fields.

The fundamental work *Disquisitiones Aritmeticae* published by Gauss in 1801 marked a new era for the theory of numbers; see Kline [4]. Gauss introduced and made systematically use of the notion of congruence of numbers to solve algebraic equations modulo a prime number, i.e. solving the equations over a prime field. With the path breaking work of Abel and Galois in the 1820s on solutions to polynomial equations, permutation groups and finite fields composed of roots to such equations came into focus. Out of this, diophantine geometry over finite fields emerged as an important research area. In the second half of the twentieth century the subject flourished. It began with the inspired survey paper on the number of solutions of equations in finite fields published 1949 by André Weil [7], in which the four famous conjectures, known as the Weil conjectures, were formulated. The last one of the Weil conjectures was resolved in 1973 by Pierre Deligne [1], a merit rewarded with the Abel Prize in 2013.

In this paper we address some questions in diophantine geometry over finite fields which appear not to have been fully explored.

Consider a plane algebraic curve C over the real numbers given as the solution set to a polynomial equation in two real variables x and y with integer coefficients.

The questions concern what remains of C, when we solve the equation over a finite field. Specifically, we study among others the following questions:

- How does the solution set change when the underlying equation is solved over a finite field?
- How does the number of solutions change with the order of the finite field?

The questions can be posed for any choice of C. To obtain specific results we need, however, to make a specific choice. In this paper we shall examine the case of C being the unit circle defined by the circle equation

$$x^2 + y^2 = 1.$$

The paper opens with a detailed study of the circle equation over the finite field $\mathbb{F}_p$ of prime order p, and more generally over the finite field $\mathbb{F}_{p^n}$ of order p^n for any natural number $n \in \mathbb{N}$. In the main Theorem 3.1 we prove that the number of solutions N_{p^n} to the circle equation over the finite field $\mathbb{F}_{p^n}$ is given by the formula

$$N_{p^n} = p^n - \sin\left(p^n \frac{\pi}{2}\right).$$

We are aware that our formula for the number of solutions to the circle equation over finite fields can be extracted from results in [6] after some nontrivial work.

Using the formula for the number of solutions to the circle equation over a finite field $\mathbb{F}_{p^n}$, we next make a study of how the number of solutions behave as a function of p^n. In Theorem 4.2 we settle the question when the circle equation over any finite field has diagonal solutions, i.e. solutions of the form $(x, y) = (x, x)$. For the prime fields $\mathbb{F}_p$ we obtain in Theorem 4.3 a very precise answer to the question how the number N_p of solutions to the circle equation over $\mathbb{F}_p$ grows as a function of p. Certain pairs of twin primes, which we term *siamese twin primes*, play a surprising role.

2. Some basic results

Since a finite field of characteristic p has order p^n for some $n \in \mathbb{N}$, only finite fields of characteristic 2 can have even order. The following theorem contains therefore, in particular, the complete answer to the question about the number of solutions to the circle equation over a finite field of even order.

Theorem 2.1. *Over the finite field $\mathbb{F}_{p^n}$ corresponding to the prime p and the integer $n \geq 1$, the equation*

$$x^{p^k} + y^{p^k} = 1$$

has exactly p^n solutions of ordered pairs (x, y) of elements $x, y \in \mathbb{F}_{p^n}$ for any integer $k \geq 1$.

Proof. Let $k \geq 1$ be an arbitrary integer. By rewriting the binomial coefficient

$$\binom{p^k}{r} = \frac{p^k!}{r!(p^k - r)!}, \quad 1 \leq r \leq p^k - 1,$$

we get

$$p^k! = \binom{p^k}{r} r!(p^k - r)! \, .$$

Making use of the prime factorization theorem, it is easily seen that p divides $\binom{p^k}{r}$. By the binomial formula we get then

$$(x + y)^{p^k} = x^{p^k} + y^{p^k}.$$

Together with the obvious relation

$$(xy)^{p^k} = x^{p^k} y^{p^k},$$

this proves that the power map $x^{p^k} : \mathbb{F}_{p^n} \to \mathbb{F}_{p^n}$ defines an isomorphism of the finite field $\mathbb{F}_{p^n}$ onto itself.

From this follows immediately that

$$(x + y)^{p^k} = 1 \quad \text{if and only if} \quad x + y = 1.$$

Clearly this implies that for every one of the p^n elements $x \in \mathbb{F}_{p^n}$, there exists a unique element $y \in \mathbb{F}_{p^n}$ such that

$$x^{p^k} + y^{p^k} = (x + y)^{p^k} = 1.$$

This proves that the equation $x^{p^k} + y^{p^k} = 1$ has exactly p^n solutions. □

Remark. There are p^{2n} ordered pairs (x, y) of elements in $\mathbb{F}_{p^n}$ and of these only p^n satisfy the equation $x^p + y^p = 1$.

The following special case of Theorem 2.1 provides as mentioned the number of solutions to the circle equation over all finite fields of even order.

Corollary 2.1. *Over the finite field $\mathbb{F}_{2^n}$ corresponding to the prime 2 and the integer $n \geq 1$, the circle equation*

$$x^2 + y^2 = 1$$

has exactly 2^n solutions of ordered pairs (x, y) of elements $x, y \in \mathbb{F}_{2^n}$.

Further on the number of solutions to the circle equation over $\mathbb{F}_p$ we have the following result.

Theorem 2.2. *Solutions to the circle equation*

$$x^2 + y^2 = 1$$

over the finite field $\mathbb{F}_p$ for p odd comes in multiples of four.

Proof. For any odd prime p, you always have the four solutions $(1, 0)$, $(0, 1)$, $(p-1, 0)$ and $(0, p-1)$. Suppose now that $(x, y) = (a, b)$, $1 \leq a, b \leq (p-1)/2$, is a solution. Then $(a, -b) = (a, p - b)$, $(-a, b) = (p - a, b)$ and $(-a, -b) = (p - a, p - b)$ are also solutions. This completes the proof. □

In the table below, we display for each of the primes $p = 2, 3, 5, 7, 11, 13$, the set of all ordered pairs (x, y) of elements in the prime field $\mathbb{F}_p$ that constitutes the set of solutions and the number N_p of solutions to the circle equation over $\mathbb{F}_p$.

TABLE 1. Solutions for $p = 2, 3, 5, 7, 11, 13$.

p	Solutions to $x^2 + y^2 = 1$	N_p
2	$(0,1), (1,0)$	2
3	$(0,1), (1,0), (0,2), (2,0)$	4
5	$(0,1), (1,0), (0,4), (4,0)$	4
7	$(0,1)$,$(0,6)$,$(1,0)$,$(2,2)$,$(2,5)$,$(5,2)$,$(5,5)$,$(6,0)$	8
11	$(0,1), (0,10), (1,0)$,$(3,5), (3,6), (5,3)$, $(5,8)$,$(6,3), (6,8), (8,5), (8,6), (10,0)$	12
13	$(0,1), (0,12), (1,0), (2,6), (2,7), (6,2)$, $(6,11), (7,2), (7,11), (11,6), (11,7), (12,0)$	12

3. Solutions to the circle equation over a finite field

In this section we extend the result for the prime 2 in Corollary 2.1 to include also the odd primes. The formula we present in Theorem 3.1 for the number of solutions to the circle equation over a finite field of odd characteristic can with some work be deduced from more general results on solutions to quadratic forms over finite fields developed by Lidl and Niederreiter in [6]. We offer, however, a self-contained direct proof of the formula.

Theorem 3.1. *For any finite field $\mathbb{F}_{p^n}$ of characteristic p, the number of solutions to the circle equation*

$$x^2 + y^2 = 1$$

over $\mathbb{F}_{p^n}$ is given by the formula

$$N_{p^n} = p^n - \sin\left(p^n \frac{\pi}{2}\right).$$

Proof. For $p = 2$ the result follows by Corollary 2.1. Hence it only remains to consider the case for an odd prime p. For convenience put $q = p^n$.

The multiplicative group $\mathbb{F}_q^*$ is a cyclic group of order $q-1$, say generated by the element $g \in \mathbb{F}_q^*$, see [5]. Every element in $\mathbb{F}_q^*$ is then uniquely presented as a power g^k of g, where the exponent k is counted modulo q.

We define the multiplicative homomorphism $\eta : \mathbb{F}_q^* \to \mathbb{S}$ of $\mathbb{F}_q^*$ onto the multiplicative group $\mathbb{S} = \{-1, 1\}$, by setting $\eta(c) = (-1)^k$, for $c = g^k \in \mathbb{F}_q^*$.

In the literature η is known as the *quadratic character* of $\mathbb{F}_q^*$. For convenience we set $\eta(0) = 0$.

The squaring homomorphism $x^2 : \mathbb{F}_q^* \to \mathbb{F}_q^*$ maps the element $a = g^l \in \mathbb{F}_q^*$ into $c = g^{2l} \in \mathbb{F}_q^*$. From this we conclude that $c = g^k$ is a square in $\mathbb{F}_q^*$ if and only if k is even modulo q, or equivalently, if and only if $\eta(c) = 1$. Hence there are equally many squares and non-squares in $\mathbb{F}_q^*$. From this follows immediately that

$$\sum_{c \in \mathbb{F}_q} \eta(c) = 0.$$

The number of solutions N_q can be decomposed into a sum of products of the number of solutions $N_q(x^2 = c_1)$ and $N_q(y^2 = c_2)$ to the equations $x^2 = c_1$ and $y^2 = c_2$, for $c_1, c_2 \in \mathbb{F}_q$ with $c_1 + c_2 = 1$. Precisely

$$N_q = \sum_{c_1+c_2=1} N_q(x^2 = c_1)N_q(y^2 = c_2).$$

Observing that the equation $z^2 = c$ over $\mathbb{F}_q^*$ has exactly two solutions if any, the expression for N_q can be rewritten as follows using the quadratic character

$$\begin{aligned} N_q &= \sum_{c_1+c_2=1} [1 + \eta(c_1)]\,[1 + \eta(c_2)] \\ &= \sum_{c_1+c_2=1} [1 + \eta(c_1) + \eta(c_2) + \eta(c_1)\eta(c_2)] \\ &= q + \sum_{c_1\in\mathbb{F}_q} \eta(c_1) + \sum_{c_2\in\mathbb{F}_q} \eta(c_2) + \sum_{c_1+c_2=1} \eta(c_1c_2) \\ &= q + \sum_{c\in\mathbb{F}_q} \eta\left(c(1-c)\right). \end{aligned}$$

Now using that $\eta(4) = \eta(2^2) = 1$ we can further rewrite this as

$$\begin{aligned} N_q &= q + \eta(-1) \sum_{c\in\mathbb{F}_q} \eta(4c^2 - 4c) \\ &= q + \eta(-1) \sum_{c\in\mathbb{F}_q} \eta\left((2c-1)^2 - 1\right) \\ &= q + \eta(-1) \sum_{c\in\mathbb{F}_q} \left(-1 + \left[1 + \eta\left((2c-1)^2 - 1\right)\right]\right) \\ &= q + \eta(-1)(-q) + \eta(-1) \sum_{c\in\mathbb{F}_q} \left[1 + \eta\left((2c-1)^2 - 1\right)\right]. \end{aligned}$$

By definition of the quadratic character η, the sum

$$S = \sum_{c\in\mathbb{F}_q} \left[1 + \eta\left((2c-1)^2 - 1\right)\right]$$

is the number of solutions in $\mathbb{F}_q$ to the quadratic equation

$$(2c-1)^2 - 1 = a^2,$$

which can be rewritten as

$$(2c - 1 + a)(2c - 1 - a) = 1.$$

To solve this product of two linear equations, observe that the factor

$$2c - 1 + a = \alpha$$

can be chosen arbitrarily in $\mathbb{F}_q^*$. Then necessarily

$$2c - 1 - a = \alpha^{-1}.$$

By subtraction of equations and division by 2, we get $a = 2^{-1}\left(\alpha - \alpha^{-1}\right)$.

Inserting this value for a into the expression for α yields

$$c = 2^{-1}\left[\alpha + 1 - 2^{-1}\left(\alpha - \alpha^{-1}\right)\right].$$

Since every solution to the quadratic equation in this way turns out to be uniquely determined by a choice of $\alpha \in \mathbb{F}_q^*$ and since the order of $\mathbb{F}_q^*$ is $q-1$, we conclude that the sum S has the value $S = q - 1$.

Collecting facts we get

$$N_q = q + \eta(-1)(-q) + \eta(-1)(q-1) = q - \eta(-1).$$

Now it only remains to determine the value of η on $-1 \in \mathbb{F}_q^*$, i.e. to determine whether -1 is a square, resp. a non-square in $\mathbb{F}_q^*$.

We can choose a generator g of $\mathbb{F}_q^*$ for which $g^0 = 1$, and $g^0, g^1, \ldots, g^{q-2}$ are all the elements in $\mathbb{F}_q^*$, when counting exponents for g modulo $q-1$.

The odd number $q = p^n$ has a unique representation either as $q = 4k+1$ or $q = 4k+3$, for k a non-negative integer.

Suppose $x = g^l$, $1 \leq l \leq (p^n - 1)/2$, is an element with $x^2 = g^{2l} = -1$. Then $g^{4l} = g^{2l}g^{2l} = (-1)(-1) = 1 = g^0$. Consequently

$$4l \equiv 0 \pmod{q-1}.$$

Now suppose $q = 4k+1$. Then we look for solutions to the congruence

$$4l \equiv 0 \pmod{4k}.$$

We get a solution if k divides l, and hence solutions always exist. We conclude that -1 is a square in $\mathbb{F}_q^*$ for $q = 4k+1$.

Next suppose $q = 4k+3$. Then we look for solutions to the congruence

$$4l \equiv 0 \pmod{4k+2}.$$

A solution exists only if $2k+1$ divides $2l$. Since an odd number can never be a factor in an even number, we conclude that the congruence has no solutions and hence that -1 is a non-square in $\mathbb{F}_q^*$ for $q = 4k+3$.

It follows that -1 is a square in $\mathbb{F}_q^*$ if and only if $q \equiv 1 \pmod 4$, and hence

$$\eta(-1) = \begin{cases} 1 & \text{if } q \equiv 1 \pmod 4, \\ -1 & \text{if } q \equiv 3 \pmod 4. \end{cases}$$

In conclusion we get

$$N_q = q - \sin\left(q\frac{\pi}{2}\right),$$

for any prime p, integer $n \in \mathbb{N}$ and $q = p^n$. □

4. Patterns in the number of solutions to the circle equation

Since we now have the precise number of solutions to the circle equation over $\mathbb{F}_{p^n}$, we can generalize Theorem 2.2 and prove the following

Theorem 4.1. *Let p be an odd prime and $n \geq 1$ an arbitrary integer. Then the number of solutions to the circle equation $x^2 + y^2 = 1$ over the finite field $\mathbb{F}_{p^n}$ is a multiple of four.*

Proof. The number of solutions is given by

$$N_{p^n} = p^n - \sin\left(p^n\,\frac{\pi}{2}\right).$$

Since p is an odd prime, $p^n \equiv 1 \pmod 4$ or $p^n \equiv 3 \pmod 4$. On the other hand, clearly

$$\sin\left(p^n \frac{\pi}{2}\right) = \begin{cases} 1, & p^n \equiv 1 \pmod 4, \\ -1, & p^n \equiv 3 \pmod 4. \end{cases}$$

It follows that

$$N_{p^n} \equiv 0 \pmod 4.$$

□

The following theorem settles in which finite fields the circle equation has *diagonal solutions*, i.e. solutions of the form $(x, y) = (x, x)$.

Theorem 4.2. *Let p be an odd prime.*

(1) *For an arbitrary integer $n \geq 1$, the circle equation $x^2 + y^2 = 1$ has diagonal solutions over the finite field $\mathbb{F}_{p^n}$ if and only if*

$$2^{(p^n-1)/2} \equiv 1 \pmod p.$$

(2) *There are diagonal solutions to the circle equation over the prime field $\mathbb{F}_p$ if and only if $p \equiv \pm 1 \pmod 8$.*
(3) *If there are diagonal solutions to the circle equation over a finite field $\mathbb{F}_{p^n}$, then there are exactly two diagonal solutions.*
(4) *If there are diagonal solutions to the circle equation over the prime field $\mathbb{F}_p$, then there are also diagonal solutions to the circle equation over $\mathbb{F}_{p^n}$ for all $n \geq 1$.*

Proof. Set $q = p^n$.

First suppose that $(x, y) = (a, a)$ is a diagonal solution to the circle equation over the finite field $\mathbb{F}_q$. Then $2a^2 = 1$ and hence $(a^{-1})^2 = 2$, showing that 2 is a square in $\mathbb{F}_q$. Next suppose that 2 is a square in $\mathbb{F}_q$. Then clearly 2^{-1} is also a square in $\mathbb{F}_q$. Therefore there exists an element $a \in \mathbb{F}_q$ such that $a^2 = 2^{-1}$, or equivalently, $a^2 + a^2 = 1$. We conclude that the circle equation has diagonal solutions in the finite field $\mathbb{F}_q$ if and only if 2 is a square in $\mathbb{F}_q$. Notice further that the equation $x^2 = 2^{-1}$ has exactly two solutions $\pm a$, if any.

To finish the proof of the theorem it only remains to determine for which $q = p^n$ the number 2 is a square in $\mathbb{F}_q$.

The finite field $\mathbb{F}_q$ is uniquely determined up to an isomorphism as the splitting field for the polynomial $f(x) = x^q - x$ in the polynomial ring $\mathbb{F}_p[x]$ over $\mathbb{F}_p$, see [5].

From this description follows easily that 2 is a square in $\mathbb{F}_q$ if and only if the polynomial $g(x) = x^2 - 2$ is a divisor in $f(x) = x^q - x$.

By polynomial division in $\mathbb{F}_p[x]$ we get

$$f(x) = x^q - x = h(x)(x^2 - 2) + (2^{(q-1)/2} - 1)x,$$

where

$$h(x) = x^{q-2} + 2x^{q-4} + 2^2 x^{q-6} + \cdots + 2^{(q-3)/2} x.$$

From the above follows immediately that $g(x)$ is a divisor in $f(x)$ and hence that 2 is a square in $\mathbb{F}_q$ if and only if

$$2^{(q-1)/2} \equiv 1 \pmod p.$$

For $n = 1$, i.e. for the prime field $\mathbb{F}_p$, it was known to Gauss (with complete proof) that 2 is a square in $\mathbb{F}_p$ if and only if $p \equiv \pm 1 \pmod 8$, see e.g. Davenport ([3], page 70).

For an arbitrary integer $n \geq 1$, the prime field $\mathbb{F}_p$ is a subfield of $\mathbb{F}_q$. Since the squaring map $x^2 : \mathbb{F}_q^* \to \mathbb{F}_q^*$ is a multiplicative homomorphism mapping $\mathbb{F}_p^*$ into itself, it follows that 2 is a square in $\mathbb{F}_q$ if 2 is a square in $\mathbb{F}_p$.

This completes the proof of the theorem. □

Examples. With reference to Table 1, there are no diagonal solutions to the circle equation over the prime field $\mathbb{F}_p$, for the odd primes $p = 3, 5, 11, 13$, whereas the are two diagonal solutions for the prime $p = 7$.

The finite field $\mathbb{F}_{3^2}$ can be described as the polynomial ring $\mathbb{F}_3[t]$ modulo the irreducible polynomial t^2+1. The nine elements in $\mathbb{F}_{3^2}$ are then uniquely described by the nine polynomials $x = a_0 + a_1 t$, for $a_0, a_1 \in \mathbb{F}_3$. Simple calculations show that $(x, y) = (t, t)$ and $(x, y) = (2t, 2t)$ are the two diagonal solutions to the circle equation over $\mathbb{F}_{3^2}$.

For the number of solutions to the circle equation over a prime field $\mathbb{F}_p$ we can do much better. As we shall see, certain pairs of twin primes, which we term siamese twin primes, turn out to play a special role.

Definition 4.1. A pair of twin primes p and p' for which $p \equiv 3 \pmod 4$ and $p' \equiv 1 \pmod 4$ is called a pair of *siamese twin primes.*

Our main result on the number of solutions to the circle equation in a prime field as a function of the prime can then be given the following concise formulation.

Theorem 4.3. *The number of solutions N_p to the circle equation*

$$x^2 + y^2 = 1$$

over $\mathbb{F}_p$ for odd primes, is a strictly increasing function of p in multiples of four, except in pairs of siamese twin primes p and p', where the function stagnates and $N_p = N_{p'}$.

Proof. Let $p < p'$ be a pair of odd prime numbers. It follows by Theorem 3.1 that $N_{p'} \geq N_p$ and by Theorem 4.1 that $N_{p'} \equiv N_p \pmod 4$.

Now suppose that $N_{p'} = N_p$. Then necessarily p and p' must be a pair of twin primes.

If $p \equiv 1 \pmod 4$ then $p' \equiv 3 \pmod 4$ since $p' = p + 2$, and hence

$$N_{p'} - N_p = p' - p - \sin\left(p'\frac{\pi}{2}\right) + \sin\left(p\frac{\pi}{2}\right) = 4,$$

which contradicts our assumption that $N_p = N_{p'}$.

On the other hand if $p \equiv 3 \pmod 4$ then $p' \equiv 1 \pmod 4$ and hence

$$N_{p'} - N_p = p' - p - \sin\left(p'\frac{\pi}{2}\right) + \sin\left(p\frac{\pi}{2}\right) = 0.$$

Altogether we conclude that $N_{p'} = N_p$ if and only if p and p' is a pair of siamese twin primes. □

It is a famous open question whether there are infinitely many pairs of twin primes. Promising progress has recently been made, e.g., [8] and [2], to settle the question in the affirmative.

If in the end it turns out that there are infinitely many pairs of twin primes, there may, however, still not be infinitely many pairs of siamese twin primes.

Based on computer tests, the present authors believe that there exist an infinite number of pairs of siamese twin primes.

In favour of this conjecture speaks that it is well known, see e.g. [3], that there are infinitely many prime numbers p and p' of each of the two types mentioned in Definition 4.1:

$$p \equiv 3 \pmod 4 \qquad \text{and} \qquad p' \equiv 1 \pmod 4.$$

The conjecture is also supported by the fact that the largest known[1] (at the time of writing) pair of twin primes

$$3756801695685 \cdot 2^{666669} - 1 \quad \text{and} \quad 3756801695685 \cdot 2^{666669} + 1$$

is also a pair of siamese twin primes, i.e.

$$N_{3756801695685 \cdot 2^{666669}-1} = N_{3756801695685 \cdot 2^{666669}+1}.$$

Although it is fairly easy on a modern computer to determine if a given pair of twin primes is a pair of siamese primes, it is not easy to determine how many pairs of siamese twins there exist below a given large number, say below

$$3756801695685 \cdot 2^{666669} + 1.$$

Denote by S_n the number of siamese twin primes below n. Likewise denote by T_n the number of twin primes below n. Then we conjecture that

$$\lim_{n\to\infty} \frac{T_n}{S_n} = 2.$$

The conjecture is supported by computer experiments with primes below $n = 2 \cdot 10^9$, for which number there are 6388041 pairs of twin primes and 3193559 pairs of siamese twin primes below n.

References

[1] Pierre Deligne. La conjecture de Weil. I. *Publ. Math., Inst. Hautes Étud. Sci.*, 43:273–307, 1973.

[2] Daniel Alan Goldston, János Pintz, and Cem Yalçin Yildirim. Primes in tuples iv: Density of small gaps between consecutive primes. *Acta Arithmetica*, 160(1):37–53, 2013.

[3] H.Davenport. *The Higher Arithmetic. An Introduction to the Theory of Numbers.* Dover Publications, Inc., New York, 1983.

[4] Morris Kline. *Mathematical Thought from Ancient to Modern Times.* Oxford University Press, 1972.

[5] Serge Lang. *Algebra.* Springer, 2005.

[6] Rudolf Lidl and Harald Niederreiter. *Finite Fields.* Cambridge University Press, 1997.

[7] André Weil. Numbers of solutions of equations in finite fields. *Bull. Amer. Math. Soc.*, 55:497–508, 1949.

[8] Yitang Zhang. Bounded gaps between primes. *Annals of Mathematics*, 179(3):1121–1174, 2014.

Technical University of Denmark

[1] `http://primes.utm.edu/top20/page.php?id=1#records`

Appendix C

Computing Homology

Since we were using Sage to calculate Gröbner bases, we can mention that it is rather easy to calculate the homology using Sage as well. Note that Sage implements calculation of reduced homology. Therefore the calculation

```
S = SimplicialComplex([[1,2], [2,3], [3,1]])
S.homology()
```

yields the following reduced homology groups,

$$\tilde{\mathrm{H}}_0(\mathbb{S};\mathbb{Z}) = 0, \quad \tilde{\mathrm{H}}_1(\mathbb{S};\mathbb{Z}) = \mathbb{Z}.$$

We have not considered reduced homology in this dissertation, but it is introduced in almost any textbook on homology theory, e.g., see [17] or [21]. Note that the augmentation of doing reduced homology is important to take into account when working with topological ranking. This is because, path connectedness is essential for the ranking. Everything still works, since the ranking considers codimensions and this is still counted when considered relative reduced pairs. Working with coefficients in a field instead of the ring of integers, we can use any linear algebra library that supports the field chosen. So if we choose either $\mathbb{R}$ or $\mathbb{C}$, then Matlab can be used. Maple works well for the fields $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{C}$ and $\mathbb{F}_p$ where p is a prime.

There is a notion of persistent homology, introduced by Herbert Edelsbrunner among others. The main difference is that in persistent homology one considers a filtration of simplicial complexes. This yields a homology theory which is more robust with respect to noise. Since the topology of power grids etc. is usually well-known it was not necessary to consider persistent homology here.

Appendix D

Code for Vertex Cover Calculations

For the calculations of the minimal vertex covers we used SageMath and thus Python. However, the scripts can be rather large and thus it was decided that a C++14 program should generate the scripts. Here we include a prototype of the C++14 program used to generate the scripts. First we present the output of using the C++ code to generate the Sage script for computing the cover ideal of the IEEE 14 bus system in parallel.

```
#!/usr/bin/env sage
import sys
from sage.all import *
import multiprocessing as mp
variableNames = [ x1 , x2 , x3 , x4 , x5 , x6 , x7 , x8 , x9 , x10↩
    , x11 , x12 , x13 , x14 ]

g = PolynomialRing(GF(2), variableNames, order= lex )
(x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11,x12,x13,x14) = g._first_ngens↩
   (14)

ideals = []
ideals.append(ideal(x1, x2))
ideals.append(ideal(x1, x5))
ideals.append(ideal(x1, x6))
ideals.append(ideal(x3, x4))
ideals.append(ideal(x2, x3))
ideals.append(ideal(x4, x5))
ideals.append(ideal(x2, x4))
ideals.append(ideal(x2, x5))
ideals.append(ideal(x4, x9))
ideals.append(ideal(x4, x7))
ideals.append(ideal(x7, x8))
ideals.append(ideal(x7, x9))
ideals.append(ideal(x9, x10))
ideals.append(ideal(x9, x14))
ideals.append(ideal(x14, x13))
ideals.append(ideal(x10, x11))
```

```
ideals.append(ideal(x11, x6))
ideals.append(ideal(x13, x12))
ideals.append(ideal(x6, x13))
ideals.append(ideal(x12, x6))

def idealsIntersection(A):
  cover = A.pop()
  for a in A:
    cover = cover.intersection(a)
  return cover

def mappingIdeals(ideals):
  idealer = []
  N = len(ideals)
  for w in range(0,4):
    idealer_tmp = []
    for t in range(0,(N)/4):
      if (len(ideals)>0):
        idealer_tmp.append(ideals.pop())
    if (idealer_tmp != None):
      idealer.append(idealer_tmp)
  return idealer

if __name__ ==  __main__ :
  p = mp.Pool()
  results = []
  ideals_pairs = mappingIdeals(ideals)
  p.map_async(idealsIntersection, ideals_pairs, callback=results.↩
   append)
  p.close()
  p.join()
  coverideal = results[0].pop()
  for I in results[0]:
    coverideal = coverideal.intersection(I)
  for I in coverideal.gens():
    print I
```

The following is the CMakeLists.txt file used to compile the code.

```
cmake_minimum_required(VERSION 3.3)
project(SageGenerator)
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++14 -w")
set(SOURCE_FILES main.cpp)
add_executable(SageGenerator ${SOURCE_FILES})
```

The main.cpp file is given below.

```
#include <iostream>
#include <stdio.h>
#include <vector>
#include <utility>
#include <algorithm>
#include <iterator>
#include <ctime>
#include <future>
#include <thread>
#include <fstream>

using namespace std;

struct Vertex {
    string name;
};

struct Edge {
    vector<Vertex> vertices;
};

struct Hypergraph {
    vector<Vertex> vertices;
    vector<Edge> edges;
};

Hypergraph generateHypergraph(int num_vertices) {
    Hypergraph hg;

    for (int i=0; i<num_vertices; i++) {
        Vertex v; v.name = "x" + to_string(i) + "";
        //cout << v.name << endl;
        hg.vertices.push_back(v);
    }

    for (int i = 0; i<num_vertices; i++) {
        int num_j;
        if (num_vertices % 2 == 0) {
            num_j = num_vertices/2;
        } else {
            num_j = (num_vertices-1)/2;
        }
        for (int j=0; j<num_j; j++) {
            if (i!=j ) {
                Edge e;
                e.vertices.push_back(hg.vertices[i]);
                e.vertices.push_back(hg.vertices[j]);
```

```
48                  hg.edges.push_back(e);
49              }
50          }
51      }
52      return hg;
53  }
54
55  template <typename T>
56  T randomFrom(const T min, const T max)
57  {
58      static std::random_device rdev;
59      static std::default_random_engine re(rdev());
60      typedef typename std::conditional<
61          std::is_floating_point<T>::value,
62          std::uniform_real_distribution<T>,
63          std::uniform_int_distribution<T>>::type dist_type;
64      dist_type uni(min, max);
65      return static_cast<T>(uni(re));
66  }
67
68  Hypergraph generateRandomHypergraph(int num_vertices) {
69      Hypergraph hg;
70      int kounter = 0;
71      for (int i=0; i<num_vertices; i++) {
72          Vertex v; v.name = "x" + to_string(i) + "";
73          hg.vertices.push_back(v);
74      }
75
76      for (int i = 0; i<num_vertices; i++) {
77          int num_j;
78          if (num_vertices % 2 == 0) {
79              num_j = num_vertices/2;
80          } else {
81              num_j = (num_vertices-1)/2;
82          }
83          for (int j=0; j<num_j; j++) {
84              if (i!=j ) {
85                  Edge e;
86                  e.vertices.push_back(hg.vertices[i]);
87                  e.vertices.push_back(hg.vertices[j]);
88                  if (randomFrom(0,1) == 1) {
89                      hg.edges.push_back(e);
90                  } else {
91                      kounter++;
92                  }
93              }
94          }
95      }
```

```
96      cout << "Number of edges removed from complete graph: " << ↩
        kounter << endl;
97      return hg;
98  }
99
100 /**
101     Build sage scripts to run.
102 */
103 string Sage() {
104     string sage = "#!/usr/bin/env sage\nimport sys\nfrom sage.all ↩
        import *\n";
105     sage += "import multiprocessing as mp\n";
106     return sage;
107 }
108
109 string SageIdealSort() {
110     string sage = "\n\n";
111     sage += "for i,id in enumerate(ideals):\n\t";
112     sage += "id = ideal(sorted(id.gens(), reverse=True))\n\t";
113     sage += "ideals[i] = id\n";
114
115     //update this to work on any number of generators.
116     sage += "ideals = sorted(ideals,key=lambda I: (str(I.gens()↩
        [0]).lower(),str(I.gens()[1]).lower()))\n";
117
118     return sage;
119 }
120
121 string SageSingle() {
122     string sage = "coverideal = ideals.pop()\n";
123     sage += "for I in ideals:\n\tcoverideal = coverideal.↩
        intersection(I)\nprint coverideal";
124     return sage;
125 }
126
127 string SageParallel() {
128     string sage = "def idealsIntersection(A):\n\t";
129     sage += "cover = A.pop()\n\t";
130     sage += "for a in A:\n\t\t";
131     sage += "cover = cover.intersection(a)\n\t";
132     sage += "return cover\n\n";
133
134     sage += "def mappingIdeals(ideals):\n\t";
135     sage += "idealer = []\n\t";
136     sage += "N = len(ideals)\n\t";
137     sage += "for w in range(0,4):\n\t\t";
138     sage += "idealer_tmp = []\n\t\t";
139     sage += "for t in range(0,(N)/4):\n\t\t\t";
140     sage += "if (len(ideals)>0):\n\t\t\t\t";
```

```
141     sage += "idealer_tmp.append(ideals.pop())\n\t\t";
142     sage += "if (idealer_tmp != None):\n\t\t\t";
143     sage += "idealer.append(idealer_tmp)\n\t";
144     sage += "return idealer\n\n\n";
145 
146     sage += "if __name__ ==  __main__ :\n\t";
147     sage += "p = mp.Pool()\n\t";
148     sage += "results = []\n\t";
149     sage += "ideals_pairs = mappingIdeals(ideals)\n\t";
150     sage += "p.map_async(idealsIntersection, ideals_pairs, ↩
    callback=results.append)\n\t";
151     sage += "p.close()\n\t";
152     sage += "p.join()\n\t";
153 
154     sage += "coverideal = results[0].pop()\n\t";
155     sage += "for I in results[0]:\n\t\t";
156     sage += "coverideal = coverideal.intersection(I)\n\t";
157     return sage;
158 }
159 
160 string SageRandomListOfIdeals() {
161     string sage = "";
162     int N = 10;
163     vector<string> list_variables;
164 
165     sage += "variableNames = [";
166     string variables = "";
167     for (int i = 0; i < N; i++) {
168         string name = "x" + to_string(i) + "";
169         list_variables.push_back(name);
170         variables += name + ",";
171         sage += " " + name + " ,";
172     }
173 
174     sage.pop_back();
175     variables.pop_back();
176 
177     sage += "]\n\n";
178     sage += "g = PolynomialRing(GF(2), variableNames, order= lex )↩
    \n";
179     sage += "(";
180     sage += variables;
181     sage += ") = g._first_ngens(" + to_string(list_variables.size↩
    ()) + ")\n\n";
182     sage += "ideals = []\n";
183 
184     int num_polys = 10;
185     for (int i=0; i<num_polys; i++) {
186         int degree = randomFrom(2,5);
```

```
187         int num_terms = randomFrom(2,4);
188         sage += "ideals.append(ideal(g.random_element(" + to_string(degree) + ", " + to_string(num_terms) + "), ";
189         sage += "g.random_element(" + to_string(degree) + ", " + to_string(num_terms) + ")))\n";
190     }
191     return sage;
192 }
193
194 string SageRandomIdeals() {
195     string sage = Sage();
196     sage += SageRandomListOfIdeals();
197     sage += "\n\n";
198     sage += SageIdealSort();
199     sage += "\n\n\n";
200     sage += SageParallel();
201     return sage;
202 }
203
204 string SageRandomIdealsSingle() {
205     string sage = Sage();
206     sage += SageRandomListOfIdeals();
207     sage += "\n\n";
208     sage += SageIdealSort();
209     sage += "\n\n\n";
210     sage += SageSingle();
211     return sage;
212 }
213
214 string SageBasic(Hypergraph hg) {
215     string sage = Sage();
216
217     //Build variable names.
218     sage += "variableNames = [";
219     string variables = "";
220     for (Vertex v : hg.vertices) {
221         //cout << v.name << endl;
222         variables += v.name + ",";
223         sage += " " + v.name + " ,";
224     }
225
226     sage.pop_back();
227     variables.pop_back();
228     sage += "]\n\n";
229     //Construct polynomial ring F_2[x0,x1,...,xn] and variables
230     sage += "g = PolynomialRing(GF(2), variableNames, order= lex )\n";
231     sage += "(";
232     sage += variables;
```

```
233     sage += ") = g._first_ngens(" + to_string(hg.vertices.size()) ↩
        + ")\n\n";
234     sage += "ideals = []\n";
235     for (Edge e : hg.edges) {
236         sage += "ideals.append(ideal(" + e.vertices[0].name + ", "↩
        + e.vertices[1].name + "))\n";
237     }
238 
239     sage += "\n\n";
240     sage += SageIdealSort();
241     return sage;
242 }
243 
244 string SageMathBuilderParallel(Hypergraph hg) {
245     string sage = SageBasic(hg);
246     sage += "def idealIntersection(A):\n\t";
247     sage += "return A[0].intersection(A[1])\n\t\n";
248     sage += "def mapIdeals(ideals):\n\t";
249     sage += "ideals_par = []\n\t";
250     //sage += "while (len(ideals) >= 1):\n\t\t";
251     sage += "for t in range(0,len(ideals)-1):\n\t\t";
252     sage += "if (len(ideals)>=2):\n\t\t\t";
253     sage += "I = ideals.pop()\n\t\t\t";
254     sage += "J = ideals.pop()\n\t\t\t";
255     sage += "ideals_par.append([I,J])\n\t";
256     //#ideals.append(p.apply_async(idealIntersection,[I,J]))
257     sage += "return ideals_par\n";
258     sage += "\n\n";
259 
260     sage += "if __name__ ==  __main__ :\n\t";
261     sage += "while ( len(ideals)!=1 ):\n\t\t";
262     sage += "p = mp.Pool()\n\t\t";
263     sage += "results = []\n\t\t";
264     sage += "ideals_par = mapIdeals(ideals)\n\t\t";
265     sage += "p.map_async(idealIntersection, ideals_par, callback=↩
    results.append)\n\t\t";
266     sage += "p.close()\n\t\t";
267     sage += "p.join()\n\t\t";
268 
269     //sage += "ideals = results[1]\n\n\t";
270     sage += "for I in results[0]:\n\t\t\t";
271     sage += "ideals.append(I)\n\t\t\t";
272 
273     sage += "\n\t";
274 
275     sage += "coverIdeal = ideals[0]\n\t";
276     //sage += "for ci in coverIdeal.gens():\n\t\t";
277     //sage += "print ci\n";
278     sage += "print coverIdeal\n";
```

```
279     //sage += "for I in ideals:\n\tcoverideal = coverideal.intersection(I)\nprint coverideal";
280     return sage;
281 }
282
283 string SageMathBuilderParallelVersion2(Hypergraph hg) {
284     string sage = SageBasic(hg);
285     sage += "def idealsIntersection(A):\n\t";
286     sage += "cover = A.pop()\n\t";
287     sage += "for a in A:\n\t\t";
288     sage += "cover = cover.intersection(a)\n\t";
289     sage += "return cover\n\n";
290
291     sage += "def mappingIdeals(ideals):\n\t";
292     sage += "idealer = []\n\t";
293     sage += "N = len(ideals)\n\t";
294     sage += "for w in range(0,4):\n\t\t";
295     sage += "idealer_tmp = []\n\t\t";
296     sage += "for t in range(0,(N)/4):\n\t\t\t";
297     sage += "if (len(ideals)>0):\n\t\t\t\t";
298     sage += "idealer_tmp.append(ideals.pop())\n\t\t";
299     sage += "if (idealer_tmp != None):\n\t\t\t";
300     sage += "idealer.append(idealer_tmp)\n\t";
301     sage += "return idealer\n\n\n";
302
303     sage += "if __name__ ==  __main__ :\n\t";
304     sage += "p = mp.Pool()\n\t";
305     sage += "results = []\n\t";
306     sage += "ideals_pairs = mappingIdeals(ideals)\n\t";
307     sage += "p.map_async(idealsIntersection, ideals_pairs, callback=results.append)\n\t";
308     sage += "p.close()\n\t";
309     sage += "p.join()\n\t";
310
311     sage += "coverideal = results[0].pop()\n\t";
312     sage += "for I in results[0]:\n\t\t";
313     sage += "coverideal = coverideal.intersection(I)\n\t";
314     sage += "print coverideal\n\t";
315
316     return sage;
317 }
318
319 string SageMathBuilder(Hypergraph hg) {
320
321     string sage = SageBasic(hg);
322     sage += SageSingle();
323     //cout << sage << endl;
324     return sage;
325 }
```

```
326
327
328
329 int main(int argc, char* argv[]) {
330
331     int num = 20;
332     if (argc > 1)
333         num = atoi(argv[1]);
334     cout << "Argument: " << num << endl;
335
336     auto hg = generateRandomHypergraph(num);
337     //auto hg = generateHypergraph(num);
338
339     string sage = SageMathBuilderParallel(hg);
340     //string sage = SageMathBuilderParallelVersion2(hg);
341     //string sage = SageRandomIdeals();
342
343     ofstream myfile;
344     myfile.open ("parallel.sage");
345     myfile << sage;
346     myfile.close();
347
348     std::system("chmod +x parallel.sage");
349
350     ofstream parV2file;
351     parV2file.open("parV2.sage");
352     parV2file << SageMathBuilderParallelVersion2(hg);
353     parV2file.close();
354     std::system("chmod +x parV2.sage");
355
356
357     ofstream singlefile;
358     singlefile.open("single.sage");
359     singlefile << SageMathBuilder(hg);
360     //singlefile << SageRandomIdealsSingle();
361     singlefile.close();
362     std::system("chmod +x single.sage");
363
364     return 0;
365 }
```

Bibliography

[1] Andreas Aabrandt and Vagn Lundsgaard Hansen. The circle equation over finite fields. *Manuscript in preparation*, 2015.

[2] Andreas Aabrandt and Vagn Lundsgaard Hansen. A note on powers in finite fields. *International Journal of Mathematical Education in Science and Technology*, to appear.

[3] Andreas Aabrandt, Vagn Lundsgaard Hansen, Bjarne Poulsen, and Chresten Træholt. Ranking entities in networks via lefschetz duality. In *IEEE European Modelling Symposium (EMS)*, pages 71–76, Oct 2014.

[4] Andreas Aabrandt, Vagn Lundsgaard Hansen, and Chresten Træholt. Topological rankings in communication networks. *International Journal of Simulation Systems, Science and Technology*, 16, 2016.

[5] Richard Bellman. *Dynamic Programming*. Princeton University Press, 1957.

[6] Richard Bellman and Robert Kalaba. *Dynamic Programming and Modern Control Theory*. Academic Press, 1965.

[7] Dennis J. Brueni and Lenwood S. Heath. The pmu placement problem. *SIAM Journal on Discrete Mathematics*, 19(3):744–761, 2005.

[8] Bruno Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenrings nach einem nulldimensionalen Polynomideal*. PhD thesis, Universität Innsbruck, 1965.

[9] David Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 3rd edition, 2007.

[10] Harold Davenport. *The Higher Arithmetic. An Introduction to the Theory of Numbers*. Dover Publications, Inc., New York, 1983.

[11] Samuel Eilenberg and Saunders Mac Lane. General theory of natural equivalences. *Transactions of the American Mathematical Society*, 58:231–294, 1945.

[12] Samuel Eilenberg and Norman Steenrod. *Foundations of Algebraic Topology*. Princeton University Press, 1952.

[13] David Eisenbud and Joe Harris. *The Geometry of Schemes*. Springer, 2000.

[14] Euclid and Thomas L. Heath. *The Thirteen Books of Euclid's Elements, Vol. 1, 2 and 3*. Dover Publications, Incorporated, 1956.

[15] Massimo Franceschet. Pagerank: Standing on the shoulders of giants. *Commun. ACM*, 54(6):92–101, June 2011.

[16] Alexander Grothendieck. Sur quelques points d'algèbre homologique, i. *Tohoku Math. J. (2)*, 9(2):119–221, 1957.

[17] Allen Hatcher. *Algebraic Topology*. Cambridge University Press, 2000.

[18] Jean-Claude Hausmann. *Mod Two Homology and Cohomology*. Online copy, 2016.

[19] Peter J. Hilton and Urs Stammbach. *Homological algebra*. Springer, 1997.

[20] Michael Jünger, Thomas M. Liebling, Denis Naddef, George L. Nemhauser, William R. Pulleyblank, Gerhard Reinelt, Giovanni Rinaldi, and Laurence A. Wolsey. *50 Years of Integer Programming 1958-2008*. Springer, 2010.

[21] Saunders Mac Lane. *Homology*. Springer, 1963.

[22] Saunders Mac Lane. *Categories for the working mathematician*. Springer, 1998.

[23] Serge Lang. *Algebra*. Springer, 2002.

[24] Solomon Lefschetz. *Applications of Algebraic Topology*. Springer-Verlag, 1975.

[25] Rudolf Lidl and Harald Niederreiter. *Finite Fields*. Cambridge University Press, 1997.

[26] W. Frank Moore, Mark Rogers, and Sean Sather-Wagstaff. Monomial ideals and their decompositions. Unpublished Manuscript, Jan 2015.

[27] Cameron Nowzari, Victor M. Preciado, and George J. Pappas. *IEEE Control Systems Magazine*, 36(1):26–46, February 2016.

[28] Grisha Perelman. The entropy formula for the ricci flow and its geometric applications. *arXiv:math/0211159*, 2002.

[29] Grisha Perelman. Finite extinction time for the solutions to the ricci flow on certain three-manifolds. *arXiv:math/0307245*, 2002.

[30] Grisha Perelman. Ricci flow with surgery on three-manifolds. *arXiv:math/0303109*, 2003.

[31] E. Spanier. *Algebraic Topology*. McGraw-Hill, 1966.

[32] Bernd Sturmfels. *Solving Systems of Polynomial Equations*. American Mathematical Society, 2002.

[33] Ian Stweart. *Galois Theory*. Chapman & Hall/CRC, 2004.

[34] Zhifang Wang, Robert J. Thomas, and Anna Scaglione. Generating random topology power grids. In *Hawaii International Conference on System Sciences, Proceedings of the 41st Annual*, pages 183–183, Jan 2008.

www.elektro.dtu.dk

Department of Electrical Engineering
Center for Electric Power and Energy
Technical University of Denmark
Ørsteds Plads
Building 348
DK-2800 Kgs. Lyngby
Denmark
Tel: (+45) 45 25 38 00
Fax: (+45) 45 93 16 34
Email: info@elektro.dtu.dk